LOAN DOCUMENT

PHOTOGRAPH THIS SHEET

AD-A241 678

DTIC ACCESSION NUMBER

LEVEL

INVENTORY

1

A Descriptive Evaluation of Software Sizing

DOCUMENT IDENTIFICATION

Sep 87

Approved for public release; Distribution Unlimited

DISTRIBUTION STATEMENT

ACCESSION FOR		
NTIS	GRA&I	☒
DTIC	TRAC	☐
UNANNOUNCED		☐
JUSTIFICATION		
BY		
DISTRIBUTION/		
AVAILABILITY CODES		
DISTRIBUTION	AVAILABILITY AND/OR SPECIAL	
A-1		

DISTRIBUTION STAMP

INSPECTED 4

DATE ACCESSIONED

STATEMENT A PER TELECON
LESLEE CRAWFORD AF COST CTR (ISO)
ARLINGTON, VA 22202
NWW 10/23/91

DATE RETURNED

91-13796

91 10 22 082

DATE RECEIVED IN DTIC

REGISTERED OR CERTIFIED NUMBER

PHOTOGRAPH THIS SHEET AND RETURN TO DTIC-FDAC

HANDLE WITH CARE

DTIC FORM 70A JUN 90

DOCUMENT PROCESSING SHEET

PREVIOUS EDITIONS MAY BE USED UNTIL STOCK IS EXHAUSTED.

LOAN DOCUMENT

AD-A241 678

Prepared For:
Headquarters USAF
Air Force Cost Center (AFCCE)
The Pentagon
Washington, D.C. 20330-5018

A Descriptive Evaluation of Software Sizing Models

September 1987

DACS

Prepared By:
Data & Analysis Center for Software
RADC/COED
Griffiss AFB, New York 13441

The Data Analysis Center for Software is a DoD Information Analysis Center operated by IIT Research Institute, Rome, NY

A DESCRIPTIVE EVALUATION OF SOFTWARE SIZING MODELS

September 1987

Prepared For:

Headquarters USAF/Air Force Cost Center (AFCCE)
Washington, D.C. 20330-5018

Prepared By:

IIT Research Institute
4550 Forbes Boulevard, Suite 300
Lanham, Maryland 20706-4324

TABLE OF CONTENTS

TABLE OF CONTENTS (Continued)

TABLE OF CONTENTS (Continued)

LIST OF ILLUSTRATIONS

TABLE OF CONTENTS (Continued)

TABLE OF CONTENTS (Continued)

GLOSSARY OF TERMS

AFB	Air Force Base
AFCCPC	Air Force Communication Computer Programming Center
AMS	Automated Measurement System
ASSET-R	Analytical Software Size Estimation Technique - Real-Time
BYL	Before You Leap
CATSS	Cartographic Applications for Tactical and Strategic Systems
CCM	Cross-Country Movement function
CD	Cockpit Display function
CDR	Critical design review
CEI	Computer Economics, Inc.
CEIS	CEI Sizer
COCOMO	COnstructive COst MOdel
CSC	Computer Software Component
CSCI	Computer Software Configuration Item
DBMS	Data Base Management System
DET	Date element type
DoD	Department of Defense
DSI	Delivered source instructions
ESD	Electronic Systems Division
FAA	Federal Aviation Administration
FC	Function point count
FP	Function points
FPA	Function Point Analysis
FTR	(Logical) file types referenced
HOL	High-order language
IDA	Institute for Defense Analysis
IITRI	IIT Research Institute
MIS	Management Information System
MLI	Machine-level instructions
Nav	Navigation function
PC	Processing complexity
PCA	Processing complexity adjustment
PDR	Preliminary design review
PERT	Program Estimating and Reporting Tool
POC	Point-of-contact
QSM	Quantitative Software Managment, Inc.
QSM (FL)	QSM Fuzzy Logic
QSM (FP)	QSM Function Points
QSM (SC)	QSM Standard Components Sizing
RCI	Reifer Consultants, Inc.
RFP	Request for Proposal
R/S	Reconnaissance/Surveillance function
SACDIN	FAA's Advanced Automation System
SAP	NASA's Source Analyzer Program
SLOC	Source lines of code
SPQR	Software Productivity, Quality, and Reliability
SPR	Software Productivity Research, Inc.
SRR	Software requirements review
SSA	Software Sizing Analyzer
SSCAG	Space Systems Cost Analysis Group

GLOSSARY OF TERMS (Continued)

SSM	Software Sizing Model
TA	Threat Avoidance function
TASC	The Analytic Sciences Corporation

PREFACE

This document provides a comprehensive review and critique of software sizing models. All of the models and techniques described in this report will estimate project size in terms of source lines of code (SLOC).

The initial focus of the report is on fourteen automated and non-automated sizing techniques, which are grouped into categories that illustrate general approaches to sizing. Of the fourteen models initially reviewed, nine models were made the subject of a more detailed analysis. To gain additional insight into the strengths and weaknesses of each model, those that were accessible were exercised and applied to a developed software system, the CATSS Sensitivity Model. A detailed description of the sizing exercise, results, and conclusions are included in this evaluation.

This special study is one of the products of the Data and Analysis Center for Software (DACS). The DACS is an information analysis center operated by IIT Research Institute (IITRI) for the Rome Air Development Center (RADC). This study was performed for the Headquarters USAF/Air Force Cost Center (AFCCE). The majority of the research reported in this review was performed by IITRI personnel at the Maryland Technology Center.

1.0 INTRODUCTION

1.1 BACKGROUND

Statistics have shown that the proportion of weapons system and MIS costs attributable to software is reaching 70% of the total acquisition and sustainment costs. Further, by the year 1990, software costs are projected to consume 10% of the entire DoD budget. These increased software costs, however, reflect the evolution of vastly improved capabilities for automated systems, with software changes allowing major system upgrades and error corrections to be accomplished at a fraction of parallel modifications implemented via hardware. With the operational readiness of the Air Force's automated systems so highly dependent upon reliable software, it is essential to accurately predict the level of resources, in both manpower and dollars, required to produce quality software for newly developing systems, as well as to effectively maintain software for currently fielded systems.

Many diverse mechanisms are presently being used in the Air Force to perform this software cost estimation. Although the underlying theories of these cost estimation techniques vary widely, nearly all of those most commonly used require the projected size of the software system as the primary input parameter from which to compute manpower effort, and subsequently, dollar figures.

Deriving an appropriate size estimate is neither straightforward nor trivial. Due to the lack of definitive information during the concept and design phases of software system development, size estimates made in those phases are characterized by uncertainty, generally resulting in estimates of very low credibility or validity. Even as systems mature in their final stages, with requirements stablized, all data inputs, outputs, and interfaces identified, and all processing functions clearly defined, the process of sizing software is still subject to a wide margin of uncertainty.

Although several efforts sponsored by both Government and industry have investigated software cost estimation models, comparatively little research on software sizing models has been pursued. Consequently, accurately projecting the size of a proposed software system remains the weakest link in the software cost estimating chain. If validated cost models are to be applied

with any degree of confidence, cost analysts must first be provided reliable means of assessing the size of the software effort involved.

Obtaining more accurate sizing estimates will in turn yield more accurate software cost estimates for future automated systems. While the results of this research effort are targeted to improving software sizing/costing prediction in the Air Force, they will be of equal utility to organizations of the other services involved in software development and long-term software support. The pervasiveness of the software sizing question is evidenced by the conclusions of a recent Army COCOMO pilot study conducted by IIT Research Institute (IITRI) under the Data and Analysis Center for Software (DACS); the findings of this study cited the inability of users to obtain accurate size estimates, due to lack of definitive information; as one of the principal barriers to implementing a standardized software cost estimating technique in the Army.

1.2 OBJECTIVE

This report provides a comprehensive review and critique of software sizing models and techniques that are available to Air Force Cost Centers. All of the models included in this evaluation will estimate project size in terms of source lines of code (SLOC). The intent is summarized by the following objectives:

- Illustrate general approaches to sizing
- Provide a consistent set of information for each automated model identified
- Identify data required to apply the models
- Clarify output results
- Rate each model according to its relevance to intended uses.

Illustrate General Approaches to Sizing

Much of the progress in the software sizing arena has been recent. A survey conducted by IITRI has identified a significant number of sizing techniques. While each represents a unique package, the underlying principles employed by some of the models are quite similar and illustrate a general approach to sizing. Those that utilize the same viable approach require

similar data input and can be applied in the same phases of the software life-cycle. Many of the same advantages and disadvantages are exhibited by similar approaches. This report will focus on a number of automated, and non-automated sizing techniques in order to provide the analyst with an overview of general methodologies applied by various techniques to estimate SLOC.

Provide A Consistent Set of Information for Each Automated Model Identified

The evaluation of specific models is limited to those automated models which were available, or at least demonstrated, to IITRI personnel who collected and assimulated model information. Of the nine (9) models included in the detailed review:

- Five (5) were available in-house to project personnel
- Two (2) were demonstrated to project personnel by the developers and were accessible through the Air Force Cost Center
- Two (2) were available in-house in the form of running prototypes.

For each automated model the following information is provided:

- source/availability/accessibility
- process description
- underlying theory (when not proprietary information)
- inputs
- outputs
- required hardware
- application area
- software life cycle phase for which the technique is most appropriate.

Information is presented in a format similar to the IDA (Institute for Defense Analyses) study on automated software cost-estimation models (IDA Paper P-1979, October 1986).

Identify Data Required to Apply the Models

The critical and optional (defaulted) data items required for use of each technique will be identified. Choosing a best method for software size estimation at a particular point in the software life cycle must be done in consonance with the type of information available to the analyst. Models such as BYL, SPQR SIZER/FP, and ASSET-R require the user to have knowledge of

external software characteristics of the system. The analyst applying the technique must have knowledge of how input screens and output reports/displays are to look. The data to be used by the application, and also a general concept of how output is to be generated should be known.

In comparison, models such as SSM and SSA can be applied successfully only be engineers thoroughly familiar with the system's functionality. Functions to be implemented within a new system are not likely to be visible from a user's perspective.

Clarify Output Results

Because the results of models vary across the techniques, outputs will also be clarified. For example, the output of Albrecht's Function Point Analysis (FPA) is given as a unitless, scalar number which is a measure of the functionality of the software, independent of lines-of-code or implementation language. A conversion table relates the function point value to a system size expressed in lines-of-code for a respective implementation language. This report will include a review of conversion factors used by non-lines-of-code methods to map the techniques' output to SLOC.

Rate Each Automated Model According to Its Relevance to Intended Uses

Choosing the appropriate sizing model(s) is a function of the information available to the analyst. A set of criteria which provides a common framework for describing the sizing models is defined. The criteria are organized so that the advantages of using one particular sizing technique versus another, as well as deficiencies, are noted.

The majority of these sizing techniques are too new to have been applied extensively throughout government and industry. To gain additional insight into the strengths and weaknesses of each technique, models to which IITRI had gained access were exercised and applied to a developed software system, the CATSS Sensitivity Model. A detailed description of the sizing exercise, results, and conclusions is included in this evaluation. The accuracy of each model is provided in terms of the size estimate produced by each model versus the actual size of the CATSS software.

The accuracy of the models for a range of applications will depend upon:

- where in the life-cycle the model is applied
- level and depth of information available for the software system
- understanding of the model by the developing organization
- familiarity with the specific software application area.

Only a certain level of accuracy and precision is possible at the early phases of the life-cycle where the level of knowledge for what the software is to do is minimum. Any sizing technique cannot be expected to compensate for a lack of understanding of a software job to be done.

1.3 REPORT OUTLINE

The guiding principle for model selection for this paper was availability of the models to the Air Force as well as consideration that the models be automated. Section 2.0, General Sizing Approaches, however includes overviews of several non-automated and semi-automated techniques in order to provide insight into many of the issues involved with software sizing. The primary focus of this paper, the summary description and evaluation of automated models, is presented in Section 3.0. The format and content of this section is similar to the IDA Report on Cost Models referenced earlier. Section 4.0 is a test case study on the application of selected models to an existing system. The study describes the process of deriving model inputs, documents required, and correlation of the resulting estimates with the actual system size. Section 5.0 summarizes the findings of the model evaluation and the test case study. The appendices present detailed information about each of the models examined in this report.

2.0 GENERAL SIZING APPROACHES

There has recently been a surge of activity to develop methodologies for arriving at a size estimate. Table 2-1 lists a number of these and the time frame in which each was made available. While each represents a unique package, the underlying principles and techniques employed by some of the models are quite similar.

TABLE 2-1

SIZING MODELS

1. ASSET-R (Analytical Software Size Estimation Technique - Real Time) - 1987
2. CEIS (CEI Sizer) - 1987
3. ESD (Electronic Systems Division) Software Sizing Package - 1987
4. Feature Points - 1987
5. QSM Size Planner - 1987
6. SPQR SIZER/FP - 1987
7. BYL (Before You Leap) - 1986
8. PRICE SZ - 1986
9. SSA (Software Sizing Analyzer) - 1985
10. Curve Fitting - 1983
11. State Machines Model - 1982
12. SSM (Software Sizing Model) - 1980
13. PERT (Program Estimating and Reporting Tool) - 1979
14. Wideband Delphi Technique - early 1970's

These sizing models were grouped into categories that illustrate general approaches to sizing shown in Table 2-2. In some cases a model will implement more than one approach. An example is The QSM Size Planner which implements three separate sizing techniques and combines the weighted outputs produced by each into a single size estimate.

TABLE 2-2

GENERAL SIZING APPROACHES

• Sizing by Analogy - ESD Sizing Package - SSA - QSM Size Planner (Fuzzy Logic)	• Comparison of Project Attributes - CEIS - QSM Size Planner (Standard Components Sizing)
• Size-In-Size-Out - Wideband Delphi Technique - SSM - Curve Fitting - PERT	• Linguistic Approach - ASSET-R - State Machines Model
• Function Point Analysis - SPQR SIZER/FP - ASSET-R - BYL - QSM Size Planner (Function Points) - Feature Points	• Other Approaches - PRICE SZ

The following subsections discuss each of the general sizing approaches. and present examples of models exhibiting those approaches.

2.1 SIZING BY ANALOGY

The sizing by analogy approach involves relating the proposed project to previously developed modules and systems of similar function and environmental requirements. The approach, as applied by the ESD and SSA sizing models, is summarized by the following sequence:

1. Develop some form of size data base consisting of descriptions of previously developed software functions and the numbers of source lines each required to implement
2. Find similarities and differences between data base items and those items being estimated
3. Select those items to serve as a basis for estimation
4. Generate a size estimate.

The accuracy of estimates derived by the analogy approach depends upon the correctness of the size data and upon the validity of analogies drawn between the data base items and those items being estimated. The size data base consists of historical project data that relates a software function with the lines-of-code count it required for implementation, either in a high-order-language (HOL) or machine-level-instructions (MLI). Various implementations of the same function provide a range of sizes. To determine where in that range the size of a new function lies requires the engineer to draw comparisons between the new function and similar functions previously developed.

The analogy approach can be applied at the system level or at the function level [ARINC85]. System level estimates which are based upon existing systems with similar applications provide size estimates in gross figures. Initially, this approach may be the only viable one if the system definition is not complete at the function level. Application of the analogy approach at the function level, though more time-consuming to apply because it requires a more detailed assessment of system components, provides more refined estimates.

The methodology/process description of the ESD, SSA, and QSM Fuzzy Logic sizing models futher illustrate the analogy approach. Each is discussed in the following three subsections.

2.1.1 ESD Software Sizing Package

Scheduled for completion in September 1987, the ESD Software Sizing Package is currently under development at ESD, Hanscom AFB. The primary participants in the package's development are Captain Joseph Dean and Ms. Barbara Mentzel.

The ESD sizer consists of two primary components: 1) the ESD sizing database, and 2) an interface that enables a user to retrieve data and generate statistical reports of the selected data set.

The data in the ESD database is taken from the following sources [ESD86]:

- The Analytic Sciences Corporation (TASC)
- Space Systems Cost Analysis Group (SSCAG) Software Sizing Database published by Aerospace Corporation
- Navy Technical Report: Software Sizing and Cost Estimation (ARINC Research Publication).

Additionally, the Army Life-Cycle Software Engineering Centers (LCSECs) are currently undergoing an extensive data collection effort and has agreed to provide their data in mid-year 1987.

The ESD software sizing database resides in the Condor 1 database environment and currently contains 825 entries. Each entry identifies previously designed/developed units of code that range in size from 2 to 500,000 SLOC. Figure 2-1 shows a sample data entry. Entry descriptors include the name of the system for which the software was developed, the function it performed, and the number of source lines it took to implement in the language indicated.

```
Function: Real Time Monitor CPCI    Index: 19.5
System:   AN/SQR-19
Status:   Code
Computer: UYK-20
Word Size: 0 (bits)
Size:     8760
Language: CMS-2
```

Figure 2-1. ESD Sample Data Entry.

Data entries are grouped into approximately 105 "standard" functions which are identified by an index number. For example, the index 19.5 in the sample data entry in Figure 2-1, typifies an acoustic support function. Appendix A, Table A-3 on page A-14, lists index numbers and related function categories for the ESD database.

To obtain a size estimate, the user may select entries from the ESD sizing database by the following entry parameters:

- Index
- Language
- System name
- Function name.
- Range of SLOC
- Development status
- Development computer

Two or more of the above parameters may also be combined to define more stringent criteria for retrieving selected entries. The Condor 1 DBMS will create a temporary data file of the selected entries, sorted by SLOC. The user may then review the entries and select a subset for manipulation by a statistics program.

The statistics program is used to obtain the record count, mean, median, variance, and standard deviation of the selected data set. Using weighted averages, the program will produce a beta distribution curve that will yield the most likely value for the new function based upon historical data contained in the ESD database. Displayed to the user is a graphic representation of the range of SLOC to be expected at the confidence level (based on standard deviation) prescribed by the user.

A sample application of the ESD Software Sizing Package is described in Section 4.3.

2.1.2 SSA (Software Sizing Analyzer)

The SSA demonstrates a sizing by analogy approach. Developed by Aerospace Corporation in 1985, the IBM PC-based package consists of the sizing database and a user-interface which facilitates the use and maintenance of the database [AERO85a].

The sizing database is comprised of primarily SSCAG (Space Systems Cost Analysis Group) and Aerospace data for space systems applications. An extensive data collection effort was initiated in 1983 and is ongoing to keep entries current.

Data entries are grouped into approximately 33 "standard" functions which are identified by a keyword for quick retrieval by data processing methods. Each data entry is identified by the descriptors listed in Figure 2-2.

```
FUNCTION: COMMUNICATIONS CONTROL
SOURCE SYSTEM: LANDSAT D
DEVELOPMENT STATUS: COMP   COMPUTER: VAX
WORD SIZE (BITS): 32   PROGRAMMING COMPLEXITY (1-5): 3
SIZE IN FORTRAN SLOC:  3679  IN MACHINE LEVEL INSTRUCTIONS: 20235
```

Figure 2-2. SSA Sample Data Entry.

The SLOC entry represents SLOC in the language indicated. The complexity entry is a rating from 1 to 5 in the judgment of the data collector. Most of the entries are considered of medium (3) complexity. The significance of the status parameter is to highlight projects not yet completed for which the size parameter is estimated rather than actual.

Conversion for each data entry from SLOC to machine level instructions (MLI) is through a conversion table supplied by RCA PRICE systems [MIZE87]. Conversion factors may be modified by changing the BASIC source code which is included with the model.

To review data for a particular function, the user must input the function's platform and keyword. Platform is either G (ground) or F

(flight). Keyword is a code of up to six characters that refers to a standard function associated with each of the database entries. For example, the data entry in Figure 2-2 would be retrieved and included in a summary of cases for the 'COMM' keyword which identifies a communication application. A detailed list of SSA standard functions identifed by keyword is included in Appendix A, Table A-4 on page A-24 [AERO85b].

The final output of the SSA is a list of each entry which met the specified search criteria. The number of cases, summarized by complexity, is also provided in the format of Figure 2-3. The analyst must determine where, in the high/low range of instructions, the new function lies.

SSA is available on a limited basis for use by the U.S. Government and Aerospace Corporation.

```
REPORT OF SIZING DATA FOR 'COMM' KEYWORD SOFTWARE WITH A 'GROUND' PLATFORM
SUMMARY BY COMPLEXITY:

AVERAGE NUMBER OF MACHINE LEVEL INSTRUCTIONS FOR  12
 CASE(S)  WITH A COMPLEXITY OF  3  IS: 5364
M.L.I. RANGES ARE FROM  100  TO  20234

AVERAGE NUMBER OF MACHINE LEVEL INSTRUCTIONS FOR  8
 CASE(S)  WITH A COMPLEXITY OF  4  IS: 3647
M.L.I. RANGES ARE FROM  302  TO  8937


    1 -I
C      I
O      I
M   2 -I
P      I
L      I
E   3 -*** * * **          *
X      I
I      I
T   4 -* *  ** *
Y      I
       I
    5 -I
       I
       I---------I---------I---------I---------I---------I---------I---------I
       0         10        20        30        40        50        60        70
                      NUMBER OF M.L.I./1000
```

Figure 2-3. SSA Sample Output.

2.1.3 QSM Size Planner: Fuzzy Logic

Fuzzy Logic is one of the three sizing techniques implemented within the QSM Size Planner. A variation of the sizing by analogy basic approach, the Fuzzy Logic technique is summarized by the following 3-step process [QSM87]:

1. Select an application category
2. Select an overall size category
3. Within the overall size category, select a size range.

The historic data used in this technique is from the QSM sizing database. The database is separated into eleven application categories such as microcode/firmware, real-time, avionics, and business software. Each application category is statistically related to a size range.

Once the user selects an application category, an overall size category must be assigned to the application. Size categories range from very small to very large. Hence, in one scenario, a user may judge a new development effort to be a small business application. In another scenario, the new development effort may be judged to be a large real-time application.

The final step is to refine the overall size estimate by providing a size range within the selected category. The size range can be on the low, midlow, midhigh, or high side of the selected size category.

Figure 2-4 is a sample Fuzzy Logic report which summarizes the user selections for application, size category, and size range. QSM Data Base Statistics shown on the report are based upon the application type and size category. The size range is reflected in the report's Estimates from User Selections. The Combined Weighted Estimate is calculated from a Baysian formulation which more heavily weighs the Estimate (either QSM Data Base or User Selections) with the smaller standard deviation.

```
                    FUZZY LOGIC SIZING REPORT

Date: 12-09-1986                          Project: slim tracking tool
Time: 16:15:24

                         User Selections

         Application Type:                      Business

         Overall Size Category:                    Small

         Size Range:                              Midhigh

                                        Mean      Std Deviation
                                     ----------   -------------
    QSM Data Base Statistics            8409          1681

    Estimates from User Selections     12083          3166

    Combined Weighted Estimate          9761          1552

PgUp-Prior Page  PgDn-Next Page  Esc-Exit
```

Figure 2-4. Sample Fuzzy Logic Sizing Report.

2.2 SIZE-IN-SIZE-OUT

Size-in-size-out techniques are purely statistical in nature and require approximate size estimates as input to apply. The resultant size estimate is a refinement of the broader, inexact estimates provided as input. Approximate size values input to these models include the following:

- Size ranges
- Relative ranking data
- Estimates for a portion of the modules defined for a system
- Several independent estimates from individual experts.

Models such as SSM and Curve Fitting must be applied at the function level and require an engineer's perspective of the modularity and functionality of the system. The Wideband Delphi Technique and PERT may be applied either at the system level or at the function level.

SSM and the Wideband Delphi Technique are expert consensus mechanisms [BOEH81] that combine individual size estimates into a single size estimate. Due to possible bias, optimism, or pessimism in individual experts who provide input size estimates, it is preferable to obtain estimates from more than one expert. The following subsections describe the Wideband Delphi, SSM, Curve Fitting, and PERT techniques, which are variations of the size-in-size-out estimation approach.

2.2.1 Wideband Delphi Technique

The standard Delphi technique originated at the RAND Corporation in 1948. The alternative Wideband method was presented by Farquahar and Boehm in the early 1970's [BOEH81]. Boehm discusses the approach in Software Engineering Economics as applied to estimating man-months of effort; however, it is appropriate for any application that requires a convergence of opinion. The technique is described by the following multi-step process:

1. Coordinator presents each expert with a specification and an estimation form
2. Coordinator calls a group meeting in which the experts discuss estimation issues with the coordinator and each other
3. Experts fill out forms anonymously
4. Coordinator prepares and distributes a summary of the estimates on an iteration form similar to that shown in Figure 2-5
5. Coordinator calls a group meeting, focusing on having the experts discuss points where their estimates varied widely
6. Experts fill out forms, again anonymously, and steps 4 through 6 are iterated for as many rounds as appropriate until consensus is reached.

The benefit of anonymous estimation is that participants in the sizing exercise are not intimidated or overly influenced by more assertive members of the group.

```
Here is the range of estimates for the 1st round:

                    Your                     Median
                  estimate                  estimate
                     ^                         ^
                     |                         |
|____|____|____|__x_|____|____|____|____|_x__|____|

5         10        15        20        25        30   (KSLOC)

Please enter your estimate for the next round: _______ SLOC
```

Figure 2-5. The Iteration Form Illustrates the Analyst's Estimate as Compared to the Median Estimate for the Group.

The standard Delphi differs from the Wideband in that no group discussion takes place. Group discussion offers considerations for estimating size of the development effort that may otherwise be overlooked. Generally, discussion will filter out extreme estimates.

2.2.2 SSM (Software Sizing Model)

Developed in 1980 by Dr. George Bozoki and refined in 1987, the Software Sizing Model (SSM) is an "expert judgment" type model which estimates size of a software project based upon two key facts:

1. The qualitative sizing information available at the proposal stage is more accurate than the corresponding quantitative data.

2. Estimators can make estimates of the relative sizes of modules more reliably than they can of their absolute sizes.

The model can be employed at any point in which the user can partition the software project into modules (components, CSCIs, CSCs...) whose operational and functional characteristics are defined. The SSM general approach can be summarized by the following four steps [BOZO87]:

1. The user initially inputs the module names/descriptions to the model; at least two modules are of known size

2. SSM generates input screens customized to the user's initial inputs

3. The user provides relative ranking data for the modules by responding to queries presented on the screens

4. SSM statistically relates the ranking data provided by the user to attain the size estimates for all of the modules.

The user initially inputs the module names and a description of each to the model. The modules must include at least two modules of known size. These modules are used as reference to calculate the size of the other modules in the set. Any existing module with a known number of lines of code will serve as a reference, and does not necessarily have to be adopted into the new system being sized. SSM generates four (4) input data set screens, customized to the user's initial inputs. The input data sets are completed by the user who performs the following tasks:

- From a unique pairing of all the modules in the project, judge which is the larger of the two for each pair

- Rank the modules from largest to smallest

- Associate each module to a designated size interval

- Provide a size range for each module (lowest possible, most likely, and highest possible).

SSM statistically relates the input data sets to attain the relative order of modules. The relative sizes are mapped to the reference modules of known size in order to attain all module sizes, standard deviations, and total system size. The output of SSM will be in the unit selected for the reference modules (source lines of code, function point count...). For SLOC estimates, the candidate reference modules must be in the same source language as those modules being sized.

Output of the model is shown in Figure 2-6. A sample application of the SSM is described in Section 4.4.

DATE: 20 AUG 87
PROJECT: LANI PHASE I
FILE: TARGET

CAPS LOCK

SSM SIZE ESTIMATES

MODULE NAME	-STD DEV	EXPECTED MODULE SIZE	+STD DEV	STANDARD DEVIATION
ON-BOARD COMPUTER EMULATION	10300	12900	15500	2600
OPERATIONS PLANNING	86400	103700	121000	17300
SPACECRAFT & SYSTEM SIMULATION	85000	85000	85000	0
SPACECRAFT MONITOR & CONTROL	23800	29900	36000	6100
SYSTEM STATUS & SCHEDULE	11400	13400	15400	2000
TREND ANALYZER	12000	12000	12000	0

SYSTEM SIZE SUMMARY (*)

EXPECTED SYSTEM SIZE: 257000
STANDARD DEVIATION: 18600

CONFIDENCE LIMITS

PROBABILITY (%)	LOW	HIGH
50	244400	269500
68	238300	275600
95	219800	294200
99	201200	312800

Figure 2-6. SSM Sample Output.

2.2.3 Curve Fitting

This technique was developed by Captain Joseph Dean while at the Air Force Communication Computer Programming Center (AFCCPC), Tinker AFB, OK. In his paper [DEAN83], Dean explained the technique by providing an example. "... A project manager was assigned a new project. After receiving the project, he was able to break it down into at least 5 basic functions. He realized that he had a very good idea of the estimated lines of code for at least 3 of the 5 functions and was able to rank them from the largest to the smallest."

Example: Project CRISYS

Function	1 =	?
	2 =	22000
	3 =	?
	4 =	5500
	5 =	2500

The next step is to input the observed points into a curve-fitting model such as PRICE-D. PRICE-D proceeds to predict the unknown functions (1 and 3) by fitting the observed points (2, 4, and 5) to one of the sixteen (16) different curves of the form:

$$F(Y) = A + B * G(X).$$

The analyst will need to evaluate the predicted values and reject those that are extraneous or unsound because meaningful limits were not found, or limits were excessive. The user bases the final estimate on functions that were acceptable. The model is purely statistical in nature and is based upon the ability of the manager to rank and predict the size of the individual functions. It is currently used with good results at Tinker AFB [DEAN87].

2.2.4 PERT (Program Estimating & Reporting Tool)

Putnam is associated with the development of the PERT technique as a means to obtain an overall system size estimate [PUTN78]. The technique, described here as a stand alone tool, has been incorporated into several models including Bozoki's SSM and Putnam's SLIM cost model.

To apply the model, experts very familiar with the proposed project must provide a range of values they feel will describe the limits of the size of each module of a system. Range values are given as the least, most likely, and highest expected number of source lines per module. The expected size of a module, E_i, is found from the weighted formulation:

$$E_i = \frac{L + 4M + H}{6} \qquad \text{where} \qquad \begin{aligned} L &= \text{Least size,} \\ M &= \text{Most likely size,} \\ H &= \text{Highest size.} \end{aligned}$$

The expected size of the total system is found by summing the expected size of each component:

$$E_{tot} = \sum_{i=1}^{n} E_i$$

The standard deviation per module is based upon the range of the experts' prediction:

$$\sigma_i = \frac{H - L}{6}$$

The standard deviation of the system is calculated using the 'square root of the sum of squares' technique given by:

$$\sigma_{tot} = \left(\sum_{i=1}^{n} \sigma^2_i \right)^{.5}$$

This results in a cancelling effect of high and low standard deviations for individual modules.

2.3 FUNCTION POINT ANALYSIS (FPA)

Originally developed in 1979 by Allan Albrecht and refined in 1983, FPA measures an application by quantifying the information processing function associated with five data types: external inputs, external outputs, external inquiries, logical internal files, and interfaces. Obtaining the function point measure is accomplished in three general steps [ALBR83]:

1. Classify and count the five data types
2. Adjust for processing complexity
3. Make the function points calculation.

Each of the data types is classified within three levels of complexity - simple, average, and complex - and then counted. The counts are recorded on an appropriate worksheet, similar to the one in Figure 2-7 currently being used in the IBM organization. Each of the data types is assigned a weight for its influence on the overall program, and the initial (unadjusted) FP count is the sum of the products of

(# of elements of a given data type) * (data type weight).

This specific information processing measure is then adjusted for processing complexity, using a multiplier ranging from .65 to 1.35 (this gives an adjustment of +/- 35%) obtained from rating 14 program characteristics. The program characterists ratings are recorded on a worksheet similar to Figure 2-7 and summed to provide the processing complexity adjustment factor. The function point measure (FP) is the product of the unadjusted function point count (FC) and the processing complexity adjustment factor (PCA).

The current function point methodology differs from Albrecht's original '79 work which

- Did not separately identify and count interfaces
- Classified all data types as average complexity (instead of simple, average, and complex)
- Produced a complexity adjustment factor which would adjust the initial FP estimate by a range of +/- 25% (instead of +/- 35%).

Type ID	Description	Complexity Simple	Complexity Average	Complexity Complex	Total
IT	External Input	___ x 3 = ___	___ x 4 = ___	___ x 6 = ___	_____
OT	External Output	___ x 4 = ___	___ x 5 = ___	___ x 7 = ___	_____
FT	Logical Internal File	___ x 7 = ___	___ x10 = ___	___ x15 = ___	_____
EI	Ext Interface File	___ x 5 = ___	___ x 7 = ___	___ x10 = ___	_____
QT	External Inquiry	___ x 3 = ___	___ x 4 = ___	___ x 6 = ___	_____
FC	TOTAL UNADJUSTED FUNCTION POINTS				_____

ID	Characteristic	DI	ID	Characteristic	DI
C1	Data Communications	_____	C8	Online Update	_____
C2	Distributed Functions	_____	C9	Complex Processing	_____
C3	Performance	_____	C10	Reuseability	_____
C4	Heavily Used Configuration	_____	C11	Installation Ease	_____
C5	Transaction Rate	_____	C12	Operational Ease	_____
C6	Online Data Entry	_____	C13	Multiple Sites	_____
C7	End User Efficiency	_____	C14	Facilitate Change	_____
PC	TOTAL DEGREE OF INFLUENCE				_____

Degree of Influence (DI) Values:

- Not present, or no influence = 0
- Insignificant influence = 1
- Moderate influence = 2
- Average influence = 3
- Significant influence = 4
- Strong influence, throughout = 5

PCA Processing Complexity Adjustment = 0.65 + (0.01 X PC) = _______

FP Function Point Measure = FC X PCA = _______

Figure 2-7. Function Points Worksheet

Of the five models in this overview which utilize the general FP approach, the BYL model correlates to the current methodology just described. SPQR SIZER/FP and QSM Size Planner have greatly modified the treatment of complexity as is illustrated in sections 2.3.2 and 2.3.3, respectively. ASSET-R and Feature Points have extended the function point work done at IBM by Albrecht for application to real-time systems.

Developers of ASSET-R and Feature Points have noted that application of the Albrecht function point methodology to size scientific and real-time systems has yielded unsatisfactory results [REIF87, SPR87]. Documented studies that support this finding could not be obtained to include in this report. The basic premise is that FP estimating formulas were derived based upcn COBOL and PL/1 data processing systems developed by the IBM DP Services organization. Therefore, the code characteristics and complexity associated with real-time parallelism, synchronization, and scientific formulation are not "captured" adequately with function point parameters.

Using Function Points

Function points were developed in 1979 by Allan Albrecht as an alternative to counting/using SLOC to analyze applications development and maintenance functions, and to highlight productivity improvement opportunities. Function points relate productivity to the amount of user function delivered whereas traditionally, productivity was always measured in terms of lines of code per man-month. The traditional metric tends to penalize higher order languages and "award" unusual code expansion due to code generators, macros, and code reuse [ALBR83].

It is worth emphasizing that function points were not originally developed as a means to derive SLOC. IBM has established the use of function points as a measurement technique for internal use. There, function point data is used to directly measure efficiency (FP/Work month), quality (defects/FP), trends in productivity, and maintenance support (work hour/FP). Permanent study groups within IBM offer comprehensive documents whose purpose it is to explain how to define and count function point parameters and how to use the estimates. Appendix E provides several scurces for FP training.

Converting FP to SLOC

The function point total is a unitless measure of the functionality of the software, independent of lines of code or implementation language. Several sizing models that implement the function point general approach output function point total and a SLOC estimate for the implementation language(s). The size prediction is based upon empirical observations of the relationship between various source languages and function points.

Table 2-3 lists source languages and the number of source lines required per function point provided by SPR, RCI, and the Gordon Group for use with their respective models, SPQR, ASSET-R, and BYL [SPR86a, RCI87, GORD87]. The table serves to illustrate the relative power of languages. For example, two programs of identical function are implemented in different languages: Basic assembler and FORTRAN. The function point total for each program is the same at 2,000. Using Table 2-3, the first program implemented in assember takes 640,000 SLOC (2000 x 320). The FORTRAN program is implemented in 210,000 SLOC (2000 x 105).

SIZER/FP and BYL permit the user to "work backwards" by providing FP parameters, processing complexity, and actual size of a completed project. The source lines per function point are generated for the specific application.

As illustrated in Table 2-3, the language expansion factors supplied by vendors will vary for the same language for several reasons:

- Values were derived by statistical analyses on different datasets
- Variations in programming skills and language experience will impact software size by as much as 50% [SPR86b]
- Model inputs tend to be subjective; hence, two analysts evaluating a completed system may derive different FP parameter counts for the system's SLOC. Thus, the ratio SLOC/FP will differ.

Initially, language expansion factors exemplify typical values for an organization based on the developer's particular dataset and model. These factors may require modification after the user has run the model successively and has evaluated the estimated versus actual size. In tailoring the factors to the organization, an inconsistency in the model's output may be corrected by factor adjustment.

TABLE 2-3

SOURCE LANGUAGES AND DEFAULT EXPANSION FACTORS
IN SPQR, ASSET-R, AND BYL

	LANGUAGE	SOURCE LINES PER FUNCTION POINT		
		SPQR	ASSET-R	BYL
1.	Basic Assembler	320	400	300
2.	Macro Assembler	213		
3.	C	128	90	128
4.	ALGOL	105		105
5.	CHILL	105	106	
6.	COBOL	105	100	105
7.	FORTRAN	105	105	105
8.	Mixed Languages (Default)	105		
9.	Other Languages (Default)	105	86	
10.	PASCAL	91	70	91
11.	RPG	80		80
12.	PL/I	80	65	80
13.	MODULA 2	80		80
14.	Ada	71	72	71
15.	PROLOG	64	64	64
16.	LISP	64		64
17.	FORTH	64		64
18.	BASIC	64		64
19.	LOGO	58		58
20.	English-Based Lanaguages	53		
21.	Data Base Languages	40		
22.	Decision Support Languages	35		
23.	Statistical Languages	32		
24.	APL	32	38	32
25.	OBJECTIVE-C	27		
26.	SMALLTALK	21		25
27.	Menu-Driven Generators	16		
28.	Data Base Query Languages	13		
29.	Spread-sheet Languages	6		
30.	Graphic Icon Languages	4		
31.	CMS-2		80	
32.	JOVIAL		80	
33.	FOCUS			40
34.	ISPF			35
35.	Multiplan			10
36.	Symphony/1-2-3			9

2.3.1 BYL (Before You Leap)

Of the four models that implement the Function Point general approach to sizing, BYL correlates to the current "baseline" methodology developed by Albrecht, and presented in the previous section.

Developed in 1986 by Donald F. Gorden, BYL contains two major subsystems [GORD87]. The first subsystem implements the COnstructive COst MOdel (COCOMO) developed by Barry Boehm at TRW. The second subsystem is the sizing capability based upon FPA. A measure of the product size produced by the sizing portion of BYL is presented to the user who may then break the estimate into new, adapted, or converted lines of code for estimating effort and schedule through the BYL COCOMO implementation.

Figure 2-8 is the input screen for the BYL function point subsystem. Using a combination of up, down, left, and right arrows and enter keys, the user may position the cursor anywhere on the FPA screen capable of accepting inputs. Inputs for function counts (External input/inquiry, External output,

```
                     tm                               F8 = View Compiler Menu
Before You Leap           FUNCTION POINT ANALYSIS
 Function Count and Complexity
External Input/Inquiry     5 Simple    5 Average    8 Complex   Compiler:
External Output            3 Simple    2 Average    6 Complex   Ada
Logical Internal File      0 Simple    5 Average    1 Complex   Coefficient:
External Interface File    0 Simple    0 Average    2 Complex     1
 Processing Complexity - Degree of Influence
Data Communications        None  Insignif  Mod  Avg  Signif  Strong
Distributed Functions      None  Insignif  Mod  Avg  Signif  Strong   Function Pt.
Performance                None  Insignif  Mod  Avg  Signif  Strong   Analysis
Heavily Used Config        None  Insignif  Mod  Avg  Signif  Strong   Suggested
Transaction Rate           None  Insignif  Mod  Avg  Signif  Strong   Lines of
Online Data Entry          None  Insignif  Mod  Avg  Signif  Strong   Delivered
End User Efficiency        None  Insignif  Mod  Avg  Signif  Strong   Source
Online Update              None  Insignif  Mod  Avg  Signif  Strong   Instructions
Complex Processing         None  Insignif  Mod  Avg  Signif  Strong   (thousands):
Reuseability               None  Insignif  Mod  Avg  Signif  Strong       17.46
Installation Ease          None  Insignif  Mod  Avg  Signif  Strong
Operational Ease           None  Insignif  Mod  Avg  Signif  Strong   Function Pt.
Multiple Sites             None  Insignif  Mod  Avg  Signif  Strong   Measure:
Facilitate Change          None  Insignif  Mod  Avg  Signif  Strong     245.92
                                   F1 = HELP
Ground Support System          Calib:dft  CstDriv:dft          33MM      9Months
```

Figure 2-8. BYL Function Point Subsystem Screen

Logical internal file, and External interface file) are numeric in nature. The numeric value assigned to each function count is multiplied by a factor depending upon the user designated complexity. The External input and External inquiry function counts are combined on the same line because these two categories have the same weighting factors across the range of simple, average, and complex.

The FP estimate is adjusted by considering the processing complexity of the product to be developed. The 14 characteristics of complexity (listed in Figure 2-8) will adjust the FP count by a factor of plus or minus 35% overall.

Values for delivered source instructions (DSI) and FP measure (Note these windows in Figure 2-8) update automatically when a change is made to any of the user-modifiable values on the screen. Converting the FP measure to DSI is based upon the compiler coefficient sometimes referred to as the language expansion factor.

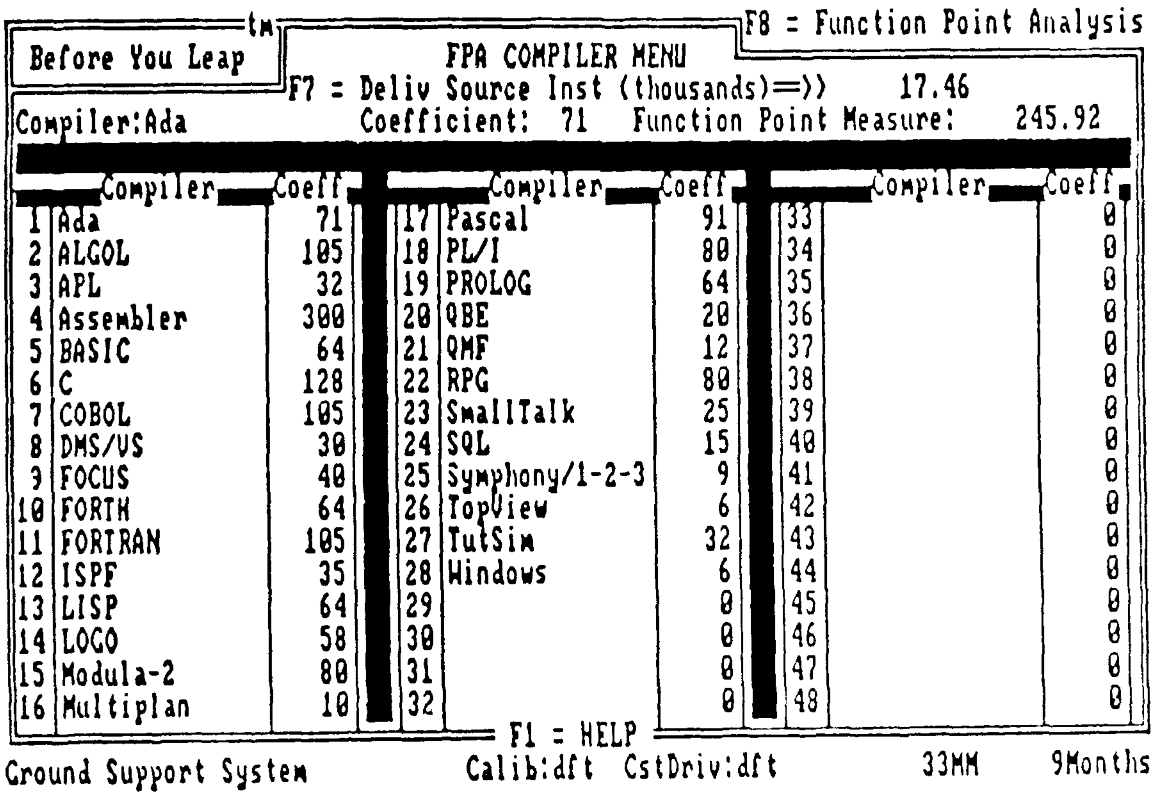


Figure 2-9. BYL Compiler Menu.

Language expansion factors are derived using the FPA Compiler Menu displayed in Figure 2-9. The user needs to evaluate at least one previous project, coded in the same language and developed in a similar environment. Working backwards, the user by provides BYL with function counts, processing complexity, and number of DSI of the previously completed project. BYL will generate a coefficient based upon the formula:

COEFF = SLOC / FP Measure.

In Figure 2-9, the currently selected compiler (i.e., Ada) is displayed near the top left of the screen with its calculated coefficient. The coefficient reflects the power of the compiler whereby the higher the coefficient, the more source instructions are required to perform a given task.

BYL output includes a summary report which demonstrates the influence of various function counts and processing complexities. A sample application of the BYL is described in Section 4.5.

2.3.2 SPQR Sizer/FP

SPR's implementation of function points is based upon Albrecht's 1979 methodology where above average and below average complexities of FP parameters are not assigned different weighting factors; that is, the initial FP total is calculated by applying the more simplistic formulation shown in Figure 2-10 [SPR86a].

Perhaps the most significant difference between SIZER/FP and Albrecht's implementation of function points is in the calculation of the complexity adjustment factor. Rather than fourteen (14) complexity adjustment factors, SIZER/FP uses three (3):

- Complexity of the problem (on a scale of 1 to 5)
- Complexity of the code (on a scale of 1 to 5)
- Complexity of the data (on a scale of 1 to 5).

According to SPR, the net effect on the function point total is a value that is within 2% of the current Albrecht methodology (described in 2.3) and obtained with less effort.

PARAMETER		**WEIGHT**		
Number of Inputs	X	4	=	______
Number of Outputs	X	5	=	______
Number of Inquiries	X	4	=	______
Number of Data Files	X	10	=	______
Number of Interfaces	X	7	=	______
		Unadjusted FP Total:		

Figure 2-10. SPR's Initial Function Point Calculation is Based Upon Albrecht's Original 1979 Methodology.

The SPR implementation of function points, developed by Capers Jones, was ~plemented within the SPQR/20 cost model in the January 1986, Version 1.1 release. Function point calculation is also offered as a stand alone capability, separate from schedule, effort, and cost estimation, in SPQR SIZER/FP.

A description of an application to size an Air Force system using the SPQR/20 model is provided in Section 4.6.

2.3.3 QSM Size Planner: Function Points

QSM's implementation of the function point sizing approach was developed after an evaluation of FP implementations by Albrecht, Capers Jones (SPR, Inc.), Xerox, and Hallmark [PUTN87]. The main data entry screen for QSM FP, shown in Figure 2-11, demonstrates how this implementation differs from the others.

From the user point-of-view, there are two primary differences. The first is that each function point parameter input is rated to five levels of complexity, ranging from very simple to highly complex (as opposed to Albrecht's three). The second distinguishing factor is that the function point count obtained is not adjusted for complexity by additional factors. Primary language is the only additional input aside from those depicted in Figure 2-11.

The weighting factors for calculating the function point total are proprietary to QSM and are not provided in this report.

FUNCTION POINTS ESTIMATES

	Very Simple	Simple	Average	Complex	Highly Complex
External Inputs	____	____2	____4	____	____
External Outputs	____2	____3	____5	____3	____
External Inquiries	____2	____4	____1	____	____
Logical Internal Files	____1	____2	____	____1	____
External Interfaces	____	____1	____1	____	____

Enter the total number of unique logical inputs that are VERY SIMPLE in this system. Logical inputs refer to both data and control information that leaves a program.

F1-HELP   F2-RESTORE DEFAULTS   F10-NEXT PAGE

Figure 2-11. QSM Data Entry Screen for Function Points.

2.3.4 Feature Points

Feature points are a superset of the Albrecht function point technique for application to real-time, embedded, military and system software [SPR87]. The approach was developed by Capers Jones at SPR, Inc. because users were not obtaining satisfactory size estimates when function points were applied to these types of systems. Table 2-6 summarizes feature point calculation. The method differs from SPR's implementation of function points (SIZER/FP) by the following characteristics:

- The weight assigned to data files is reduced
- Number of algorithms is introduced as a new parameter.

Complexity adjustment parameters are the same as those defined for SIZER/FP.

Algorithm is defined as a "(complete) set of rules which... solve a computational problem." In an example supplied by SPR, the following equations compose a single algorithm:

Staff = Size / Assignment Scope
Effort = Size / Production Rate
Schedule = Effort / Staff.

The SPR feature point method is currently undergoing testing and will be released in mid-1987. For additional information, the point of contact is Software Productivity Research, Inc. in Cambridge, Massachusetts.

ITEM		WEIGHT		
Number of Algorithms	X	6	=	______
Number of Inputs	X	4	=	______
Number of Outputs	X	5	=	______
Number of Inquiries	X	4	=	______
Number of Data Files	X	4	=	______
Number of Interfaces	X	7	=	______
Subtotal				
Complexity Adjustment (0.6 to 1.4)				
Adjusted Feature Point Total				

Figure 2-12. Feature Point Parameters.

2.4 LINGUISTIC APPROACH

The linguistic approach applies to counting the lexical symbols used in the programmatic expression of an algorithm. These lexical symbols can be found in the engineering equations of system specifications and pseudo-code of the detailed design phase, or applied after-the-fact to the source code of a completed system. Maurice Halstead initially demonstrated a fundamental relationship between the (unique) operator count and the length of the program text, stated in terms of tokens [GAFF84]. This relationship is:

$$N = n_1 \log_2 n_1 + n_2 \log_2 n_2 \quad \text{where}$$

N = number of tokens
n_1 = operator vocabulary size
n_2 = operand vocabulary size.

Examples of unique operators are READ, =, +, IF; operands are constants and variables. A string of tokens is likened to a sequence of instructions that constitute the program text. The formula was originally derived to apply to a program implementation of an algorithm or function. To derive the software length for a program with multiple functions would involve applying the software length equation, above, to each function individually, and summing the results [ALBR83].

Halstead's work provides theoretical support for the State Machines Model whose "variable count" input parameter corresponds to Halstead's n_2, operand vocabulary size [GAFF84]. Similarly, ASSET-R formulation derives size as a function of the adjusted function point count and operator/operand count.

2.4.1 State Machine Model

Britcher and Gaffney [BRIT85] have suggested that any software system should be divisible into 6 "levels", ranging from level 0, the initial program specification, through level 5, the code. Figure 2-13 illustrates the levels which represent more detailed statements of the system requirements without reference to how the requirements should be implemented within the program.

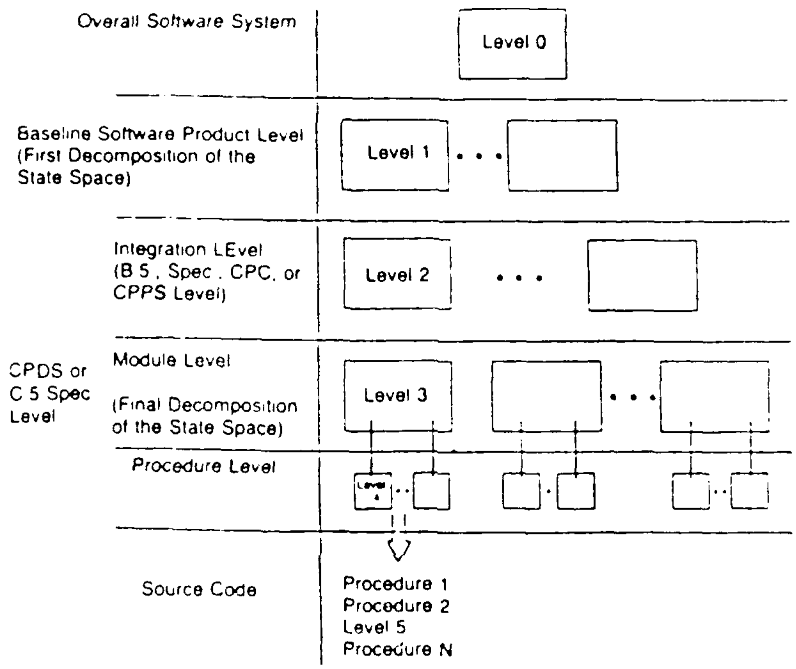


Figure 2-13. Levels of Specification for System Requirements.

Each successive level contains proportionately more "function boxes". Since any software system should have the same number of specification levels, a system with more requirements should have more boxes at each level (as opposed to "bigger" boxes at each level). The greater number of function boxes at higher levels yield more source code at the lowest level. Hence, one should be able to produce a size estimate based upon the number of boxes at a certain level, given that, on the average, each box per level contains about the same amount of function and results in about the same amount of code.

The size of any one function box is estimated by specifying the amount of data (variable count) it is to use and generate. At the procedure level the variable count (v) is applied to the following formula [GAFF84]:

$S = (4.8078/N) \times V \times \log_e V$ where

$N = 1$, if $V < 100$
$N = 2$, if $100 \le V < 1000$
$N = 3$, if $1000 \le V \le 10{,}000$ and
S = average number of assembly SLOC per box
V = variable count for a box at the procedure level.

Once the size is calculated for one box, the value is multiplied by the number of boxes at the procedure level in order to estimate system size.

In a sample application, using data from the SACDIN project (FAA's Advanced Automation System; not related to Strategic Air Command Digital Interface Network), an average value of 6 was found for variable count (V) at the procedure level [BRIT85]. Applying the formula [GAFF84], size (S) was estimated to be 52 SLOC per procedure. There were an average of 4 (level 4) procedures per (level 3) module or 208 assembly SLOC. For each (level 2) CPC there were 20 (level 3) modules per box. There were 10 (level 2) CPC's per (level 1) CPCI. Derived number of SLOC for SACDIN software are summarized for the specification levels in Table 2-4.

TABLE 2-4
DERIVED SLOC ESTIMATES FOR SACDIN SPECIFICATION LEVELS

Function, Specification, or Abstraction Level		Estimated Average No. of Assembly SLOC Per Box
No	Name/Type	
4	Procedure	52*
3	Module	208
2	Integration (b-5, CPC)	4160
1	Baseline (C-5, CPCI)	41600

*$S = (4.8078/N) \times V \times \log_e V$

Where: $N = 1$, If $V < 100$
$N = 2$, If $100 \le V < 1000$
$N = 3$, If $1000 \le V \le 10000$

Where: V = The State Machine "Variable Count" (at the Procedure Level); It corresponds to Halstead's η^*_2, the "Operand" Vocabulary Size.

The State Machine Model is a conceptual method and has not been implemented in an automated model.

2.4.2 ASSET-R (Analytical Software Size Estimation Tool - Real Time)

The ASSET-R model is a broad extension of Albrecht's function point work to real-time and scientific systems. Developed by Donald Reifer of RCI, Inc., the model also incorporates Halstead's software equations. ASSET-R extensions to Albrecht's work primarily provide a measure of the code associated with real-time performance requirements and scientific formulation. Extensions are represented by the following equation [RCI87]:

$$\text{SIZE (SLOC)} = (\text{ARCH})\,(\text{EXPF})\,((\text{LANG} * FP_A) + OOC_N)^{rf}$$

ARCH = architectural constant
EXPF = technology expansion factor
LANG = language expansion factor
FP_A = function point count (adjusted)
OOC_N = operand/operator count (normalized)
rf = reuse factor.

Several user inputs are usually required to derive each of the factors that compose the ASSET-R mathematical formulation. The exceptions are the architectural constant (ARCH) and language expansion factor (EXPF) which are each obtained from a table of values that directly correlate to the user's input. *Tables A-1 and A-2 on page A-6 in Appendix A demonstrate how these* factors are defined.

The technology expansion factor (EXPF) reflects the experience base of the user organization. Nine parameters contribute to EXPF:

- Requirements Volatility
- Database size
- Use of software tools
- Applications experience
- Programming language experience
- Degree of real-time code
- Environment experience
- Analyst capability
- Use of modern programming techniques

Formulations to determine function point count utilize different weighting factors and data types which are dependent on the type of system being sized: data processing, scientific, or real-time. Table 2-5 illustrates this concept. User FP parameter inputs to the model for data processing, scientific, and real-time systems are identical. However, depending upon the type of system, some inputs are weighted more heavily. The number of function points (FP_A) are calculated by applying the weighting factors to FP parameters and adjusting for complexity. Complexity adjustment

TABLE 2-5

ASSET-R WEIGHTING FACTORS FOR DATA PROCESSING, SCIENTIFIC, AND REAL-TIME SYSTEMS

(DATA PROCESSING) **WEIGHTING FACTORS**

FUNCTION POINT PARAMETERS	VERY LOW	LOW	NOMINAL	HIGH	VERY HIGH	EXTRA HIGH
External Inputs	4	4	4	4	5	5
External Outputs	4	4	4	4	5	5
Internal Files	6	6	8	8	8	8
External Inquiries	4	4	5	5	6	6
External Interfaces	4	4	6	6	6	6

(SCIENTIFIC) **WEIGHTING FACTORS**

FUNCTION POINT PARAMETERS	VERY LOW	LOW	NOMINAL	HIGH	VERY HIGH	EXTRA HIGH
External Inputs	4	4	4	4	4	4
External Outputs	4	4	4	4	5	5
Internal Files	8	8	8	8	8	8
Operating Modes	2	2	2	2	2	2
External Inquiries	4	4	5	5	6	6
External Interfaces	3	3	4	4	5	5

(REAL-TIME) **WEIGHTING FACTORS**

FUNCTION POINT PARAMETERS	VERY LOW	LOW	NOMINAL	HIGH	VERY HIGH	EXTRA HIGH
External Inputs	1	1	1	1	1	1
External Outputs	1	1	1	1	1	1
Internal Files	1	1	1	1	1	1
Operating Modes	1	1	1	1	1	1
Stimulus/Response Relationships	1	1	1	1	1	1
Rendezvous	1	1	1	1	1	1
External Inquiries	1	1	1	1	1	1
External Interfaces	1	1	1	1	1	1

is internal to the model and is not obtained from rating program characteristics as described in Section 2.3.

Other factors that compose the ASSET-R basic equation are operator/operand count (OOC_N) and reuse factor (rf). Derivation of the reuse factor is internal to the model. In most cases it is set to 1. However, in instances of high reuse where rf can range between .88 and .95, it can greatly impact software size because of its exponential effect.

Operator/operand counts indicate the amount of code required to implement mathematical formulation. Unfortunately, obtaining this measure is not a straightforward task. Operator/operand counts can be obtained by analyzing the engineering equations that are to be used in the developed software. Ideally, the formulation is provided as part of the system specification. Counts are obtained manually by identifying algorithms and counting the number of operators and operands in specification documents. Counts may also be obtained from the code used for simulation using an automated code analyzer. The number of basic algorithms may be used as a measure of mathematical formulation in lieu of the number of operators and operands. The type of system being sized (scientific, real-time, or data processing) will effect the emphasis that is placed on this factor.

A sample application of the ASSET-R model is described in Section 4.7.

2.5 COMPARISON OF PROJECT ATTRIBUTES

Although data collection and evaluation of completed projects serve to calibrate some models, two of the five general approaches to sizing, categorically defined, require historic project data on which to base new estimates:

- Sizing by analogy
- Comparison of project attributes

Sizing by analogy relates previously developed software to a new development effort at either the system or function level. However, system components of completed projects are assessed, relative to new development efforts, at a more detailed level of specification, within the comparison of project attributes category.

Generally, parameters are identified that correlate to the size of a new program. Attributes of a new project (i.e., required reliability, complexity, number of screens, reports, etc.) are compared with the same attributes of completed projects. Statistical analysis that correlates attributes of the new development effort to completed projects is performed to yield an estimated size for the new effort.

CEIS and QSM Standard Component Sizing demonstrate unique applications of this sizing approach.

2.5.1 C. ; (CEI Sizer)

The Computer Economics, Inc. Size (CEIS) Estimation System was initially developed during 1986 and is currently under validation; it should be released later in 1987. The CEIS software sizing model allows the code of a software task (program or project) to be estimated by comparing the attributes of the new task to the attributes of three reference tasks of known size. The basic concepts of the approach were developed by T.L. Saaty [SAAT80]. Dr. Joseph M. Lambert applied these concepts to software sizing in a paper [LAMB86] presented at the 1986 ISPA conference.

The estimation of a new task's code size is a three-step process:

- Determine the relationship between (6) project attributes
- Determine the relationship between the (3) reference tasks
- Determine the relationship of the new task to the (3) reference tasks.

At each step, a comparison is performed to weigh the importance of one attribute to another using the effect on overall lines of code as a basis for the decision. The intensity of importance is a numeric value ranging from 1 to 9 where 1 demonstrates equal importance and 9 favors one attribute to be of overwhelming importance relative to its effect on size. Relative importance is in accordance with the following definitions:

1 = Equal Importance	Two attributes or tasks contribute equally to the objective
3 = Weak Importance	Experience and judgment slightly favor one over the other
5 = Strong Importance	Experience and judgment strongly favor one over the other
7 = Demonstrated Importance	Activity is strongly favored and dominance is demonstrated in practice
9 = Absolute Importance	Evidence favoring one activity over another is overwhelming.

Intermediate values (2, 4, 6, and 8) are used when compromise is needed between the above values.

Figure 2-14 illustrates how attributes are calibrated for the CEIS application. A graphic representation of a slide potentiometer is used

```
Compare Attributes - Page 1 of 2

                          vs. RRLY            vs. SPEC            vs. RVOL

                       9 7 5 3 1 3 5 7 9   9 7 5 3 1 3 5 7 9   9 7 5 3 1 3 5 7 9
Complexity                         5                 1                     5
                       - - - - - - - - -   - - - - - - - - -   - - - - - - - - -
Peak Staff                   2                 3                   3
                       - - - - - - - - -   - - - - - - - - -   - - - - - - - - -
Technology Rating                  5                     5                 3
                       - - - - - - - - -   - - - - - - - - -   9 7 5 3 1 3 5 7 9
Reqm'ts Volatility           2                 3
                       - - - - - - - - -   9 7 5 3 1 3 5 7 9
Specification Level              3
                       9 7 5 3 1 3 5 7 9

                                            Comparison Help
                                            1 - Equal Importance
                                            3 - Weak Importance
                                            5 - Strong Importance
                                            7 - Demonstrated Importance
                                            9 - Absolute Importance
```

```
Compare Attributes - Page 2 of 2

                          vs. TECH            vs. STAF

                       9 7 5 3 1 3 5 7 9   9 7 5 3 1 3 5 7 9
Complexity                   3                           7
                       - - - - - - - - -   9 7 5 3 1 3 5 7 9
Peak Staff                 5
                       9 7 5 3 1 3 5 7 9

                                            Comparison Help
                                            1 - Equal Importance
                                            3 - Weak Importance
                                            5 - Strong Importance
                                            7 - Demonstrated Importance
                                            9 - Absolute Importance
```

Figure 2-14. CEIS Attributes Calibration.

to designate the relative importance of each attribute. If the attribute listed on the row is more important, the potentiometer is moved to the left. If the attribute listed for the column is more important, the potentiometer is moved to the right.

Table 2-6 shows the pairwise comparison matrix that is formed from the attribute calibrations designated in Figure 2-14. In the example, the upper triangle (above the main diagonal) of the matrix are the numeric ratings assigned by the user. Reciprocal values of each pairwise comparison complete the lower triangle of the matrix. Hence, when a pairwise comparison of (peak staff, technology rating) is rated by the user as 1/5, a pairwise comparison of (technology rating, peak staff) is inversely determined to be 5.

Once the data is in matrix form, eigenvalue/eigenvector analysis is performed to obtain the following two measures:

1. A relative measure of consistency of the matrix
2. Weights for each of the project attributes.

Eigenvalue/eigenvector analysis is the general technique of finding eigenvalues and associated eigenvectors of any square matrix. Given a matrix A, a scaler λ, and a vector X; *the vector* X *is an* **eigenvector** of A and the scaler λ is an **eignevalue** of A if the equation $AX = \lambda X$ is satisfied.

TABLE 2-6

CEIS COMPARISON MATRIX

	CMPL	STAF	TECH	RVOL	SPEC	RRLY
COMPLEXITY	1	7	1/3	5	1	5
Peak Staff	1/7	1	1/5	1/3	1/3	1/2
Technology Rating	3	5	1	3	5	5
Reqm'ts Volatility	1/5	3	1/3	1	1/3	1/2
Specification Level	1	3	1/5	3	1	3
Required Reliability	1/5	2	1/5	2	1/3	1

Solving the eigenvalue/eigenvector problem, the eigenvalue (λ) is used to obtain the consistency ratio. If the consistency ratio is inappropriate, then the matrix will need to be reevaluated. Otherwise, the associated eigenvector is found for λ. The eigenvector is normalized so that the sum of the components of the vector equals one. The normalized vector values are the attribute weights.

In a similar manner, weights are obtained for three reference tasks of known size and a fourth task with unknown size. The user enters the three reference tasks and their associated sizes to the model. Using the potentiometer, reference tasks are compared to one another for each of the attributes (see Figure 2-15). In the last step, the relationship of the new task to each of the reference tasks for each of the attributes is established using the potentiometer approach (see Figure 2-16).

After all these relationships have been determined, the unknown size is computed as a function of the unknown project weight, and the actual sizes of the three reference tasks. The new size estimate is displayed in the lower left window of the new task size display shown in Figure 2-16.

```
Compare Reference Tasks

                            A vs. B             B vs. C             C vs. A

                      9 7 5 3 1 3 5 7 9   9 7 5 3 1 3 5 7 9   9 7 5 3 1 3 5 7 9
Complexity                          7                 5       9
                       - - - - - - - -     - - - - - - - -     - - - - - - - -
Peak Staff                  3                   3                         5
                       - - - - - - - -     - - - - - - - -     - - - - - - - -
Technology Rating           3                   3                         5
                       - - - - - - - -     - - - - - - - -     - - - - - - - -
Reqm'ts Volatility           2                   2                      3
                       - - - - - - - -     - - - - - - - -     - - - - - - - -
Specification Level         3                   3                         5
                       - - - - - - - -     - - - - - - - -     - - - - - - - -
Required Reliability        3                   3                         5
                      9 7 5 3 1 3 5 7 9   9 7 5 3 1 3 5 7 9   9 7 5 3 1 3 5 7 9

Task Names and Sizes                              Comparison Help
            Name                   Size           1 - Equal Importance
Task A:     Situation Display      20347          3 - Weak Importance
Task B:     Radar Prediction       12200          5 - Strong Importance
Task C:     LOS Determination       8300          7 - Demonstrated Importance
                                                  9 - Absolute Importance
```

Figure 2-15. CEIS Reference Tasks Calibration.

```
Output Results to Display

                          New vs. A            New vs. B            New vs. C

                      9 7 5 3 1 3 5 7 9    9 7 5 3 1 3 5 7 9    9 7 5 3 1 3 5 7 9
Complexity              7                      5                              3
                      - - - - - - - - -    - - - - - - - - -    - - - - - - - - -
Peak Staff                    1                  3                    3
                      - - - - - - - - -    - - - - - - - - -    - - - - - - - - -
Technology Rating               3                    1                        3
                      - - - - - - - - -    - - - - - - - - -    - - - - - - - - -
Reqm'ts Volatility             2                     1                3
                      - - - - - - - - -    - - - - - - - - -    - - - - - - - - -
Specification Level             3                    1                3
                      - - - - - - - - -    - - - - - - - - -    - - - - - - - - -
Required Reliability            3                    1                3
                      9 7 5 3 1 3 5 7 9    9 7 5 3 1 3 5 7 9    9 7 5 3 1 3 5 7 9

Task Names and Sizes                                 Comparison Help
               Name                  Size            1 - Equal Importance
Task A:    Situation Displays        20347           3 - Weak Importance
Task B:    Radar Prediction          12200           5 - Strong Importance
Task C:    LOS Determination          8300           7 - Demonstrated Importance
New Task:  Map Background            12837           9 - Absolute Importance
```

Figure 2-16. CEIS New Task Size Display.

2.5.2 QSM Size Planner: Standard Components Sizing

The QSM Standard Component Sizing function computes an estimate of the size of the software system based on the expected number of components the system will contain [QSM87]. The user may elect to use his own historical data, QSM data, or a weighted combination of both as a basis for converting component estimates to source statements, and then statistically combining each estimate to yield a single best estimate.

The twelve components are listed in the summary report in Figure 2-17. In the sample report, the user inputs a range of expected values for SLOC, files, modules, and reports as well as the source language. The inputs were converted to an expected size of 10731 based upon the size and consistency of a combination of QSM data and the user's own historical data (note the weight to QSM data in Figure 2-17).

STANDARD COMPONENT SIZING REPORT

Date: 12-09-1986
Time: 16:15:33

Project: slim tracking tool
Language: Basic

Component	Low	Most Likely	High	Conf.	Weight to QSM Data	Expected SLOC	Std Dev SLOC
SLOC	5000	11000	17000	2	100	11000	3000
Obj Instr.	0	0	0	1	100	0	0
Bits	0	0	0	1	100	0	0
Bytes	0	0	0	1	100	0	0
Words	0	0	0	1	100	0	0
Files	5	10	15	2	100	1350	473 [5]
Modules	10	15	25	2	100	9342	2213
Subsystems	2	4	6	1	100	12052	3013
Screens	0	0	0	1	100	0	0
Reports	0	0	0	1	100	0	0
Interactive	0	0	0	1	100	0	0
Batch Pgms	0	0	0	1	100	0	0

Combined Weighted Solution: 10731 +- 1561

[5] This size entry has not been included in the solution process because it is a low confidence statistical outlier.

Figure 2-17. Standard Component Sizing Report.

The column marked Weight to QSM Data will automatically be set to 100 if the user has no historical data. Otherwise, the value is calculated based on the user's historical database. The user may revise the QSM calculated data weight values to any value between 10 and 100.

'Conf.' (in Figure 2-17) refers to Confidence Level. It represents the user's confidence in the component as an indicator to system size. Assigned Conf. impacts the weight which is given to a specific component. A value of 1 indicates a low confidence level, 2 is moderate, and 3 reflects high confidence in the component.

2.6 OTHER APPROACHES

Only one model in this survey could not be explicitly associated with any of the general approaches categorically defined - PRICE SZ.

The model is a black-box; little is known about the underlying techniques employed by the model. It does not implement a functin point approach but some of its quantitative inputs are similar to function point parameters.

The next section provides an overview of this model.

2.6.1 PRICE SZ

Though it is currently the most used sizing model due to its accessibility to DoD personnel, details are unknown about the Sizer's algorithm. PRICE SZ was developed during the 1980-1985 time period by a team of PRICE Systems Engineering personnel (Dr. Robert Park, William Rapp, and Carmen Rueda). Development was based mainly on PRICE software experience with some validation against RCA Moorestown Missile and Surface Radar Division projects as well as against data in literature [MANC87].

Essentially, the size estimate is a function of its quantitative inputs, summarized as follows:

> F (Output pages, Input files, Output files, Work files, Control/States, I/O Variables/Tables).

Environmental and calibration factors are used to adjust t size estimate to the developing environment and organization. A description of all PRIZE SZ inputs is provided in Appendix A [RCA85]. The range of outputs for the sizing estimate correspond to a joint variation of 10% in all of the inputs.

A sample application of PRICE SZ is described in Section 4.8.

3.0 DESCRIPTION AND EVALUATION OF MODELS

3.1 SELECTION

Availability of the models to the Air Force as well as consideration that the models be automated were the guiding principles for model selection for this paper. Given the criteria, nine products will remain the focus of this evaluation:

- ASSET-R
- CEIS
- PRICE SZ
- SPQR SIZER/FP
- SSM
- BYL
- ESD
- QSM Size Planner
- SSA

Several models that were discussed in Section 2.0 are not included in the detailed evaluation. The Wideband Delphi Technique and the State Machines Model exist only on paper. PERT is not automated as a stand-alone tool although it has been implemented in several models. Feature Points and the Curve Fitting techniques were excluded from detailed evaluation because running prototypes were not available for review.

All of the models included in the detailed evaluation were, at the least, demonstrated to IITRI personnel who collected and assimilated model information; six of the models were exercised by IITRI on a sample system (model applications are described in Section 4). Detailed information provided in this report is based upon actual use of the models or demonstrations, in addition to reference materials, and some lengthly discussion with the model developers or points of contact.

3.2 SUMMARY DESCRIPTION OF MODELS

This section contains general information about each of the models included in the evaluation. For each, the vendor or point-of-contact (POC), hardware requirements, availability, and cost are examined. Input parameters to be supplied by the user are listed, with input parameters grouped by input categories. Also included are a comparisons of techniques employed and model outputs.

3.2.1 Vendor/Point of Contact

Most of these models are undergoing continual revision as developers receive feedback from their users. For additional information about any of these models, the individual vendors/POCs (listed in Table 3-1) should be contacted.

TABLE 3-1

SIZING MODEL POINTS OF CONTACT

MODEL	VENDOR/POC
ASSET-R	Reifer Consultants, Inc. 25550 Hawthorne Blvd., Suite 208 Torrance, CA 90505-6825 (213) 373-8728
BYL	Gordon Group 1425 Koll Circle, Suite 102 San Jose, CA 95112 (408) 280-0743
CEIS	Computer Economics, Inc. 4560 Admiralty Way, Suite 109 Marina del Ray, CA 90292 (213) 827-7300
ESD	Ms. Barbara Mentzel ESD/ACCR Hanscom AFB, MA 01731-5000 (617) 377-2674 Autovon: 478-2674
PRICE SZ	RCA PRICE Systems 300 Route 38, Bldg. 146 Moorestown, NJ 08057 (609) 866-6583
QSM Size Planner	Quantitative Software Mgmt., Inc. 1057 Waverly Way McLean, VA 22102 (703) 790-0055
SPQR SIZER/FP	Software Productivity Research,Inc. 2067 Massachusetts Avenue Cambridge, MA 02140 (617) 495-0120
SSA	Mr. Gerard Heydinger SD/ACCE Los Angeles Air Force Station P.O. Box 92960 Los Angeles, CA 90009-2960 (213) 643-1772
SSM	Dr. George J. Bozoki Target Software 552 Marine Parkway #1202 Redwood City, CA 94065 (415) 592-2560

3.2.2 Hardware Requirements

Table 3-2 summarizes the hardware requirement for each of the models. All but two of the models are available on an IBM PC (or compatible). Additional details concerning hardware requirements are contained in Appendix B.

TABLE 3-2

HARDWARE REQUIREMENTS

	IBM PC	ZENITH-100	PRIME	VAX 11/780	TIME SHARE
ASSET-R	X				
BYL	X				
CEIS	X	X		X	X
ESD		X			
PRICE SZ			X		X
QSM	X				
SIZER/FP	X				
SSA	X				
SSM	X		X		C

X = Available to DoD and Commercial users
C = Available to Commercial users only

3.2.3 Availability

Table 3-3 summarizes the availability of each model to DoD and commercial users. Availability through request means that a potential user can receive the model at no cost by contacting the model POC. The model is received on a diskette that is provided by the requesting agency. Additional contractual information is contained in Appendix C.

TABLE 3-3

CONTRACTUAL ARRANGEMENTS

	PURCHASE	LEASE	TIME SHARE	REQUEST
ASSET-R		X		
BYL	X			
CEIS		X	X	
ESD				X
PRICE SZ		X	X	
QSM		X		
SIZER/FP	X			
SSA				D
SSM	X		C	

X = Available to DoD and Commercial users
D = Available to DoD users only
C = Available to Commercial users only

3.2.4 Costs

Table 3-4 shows the DoD rates for each of the models. It is clear from the table that there is a wide range in the cost of the models. Additional costs may be associated with user training, which is required for some of the models. Also, rates will vary depending upon the type of licensing agreement procured (annual, site, corporate, etc.).

TABLE 3-4

LEASE/PURCHASE RATES
(DoD)

	FIRST UNIT	EXTRA UNITS	TIME SHARE
ASSET-R Lease:	$8000/year	Negotiable	
BYL Purchase:	$950	Cost/unit decreases	
CEIS Lease:	No added cost to System-3 users		$49.25/Hour
ESD Request:	No cost, but send diskette		
PRICES SZ			$75/Hour
QSM Lease:	$9500/year	$500/copy	
SIZER/FP Purchase:	$500	Cost/unit decreases	
SSA Request:	No cost, but Controlled access		
SSM Purchase:	$1849	Cost/unit decreases	

A number of the models are associated with an automated cost model developed by the same vendor (See Table 3-5). Rates may depend upon whether a user is accessing both the sizing and the cost models or only the sizing model. For example, CEIS is available to SYSTEM-3 users at no additional costs while non-System-3 users pay a substantial fee. (Exact rates for non-System-3 users have not yet been established). Appendix C provides additional information concerning access costs.

TABLE 3-5

COST MODEL IMPLEMENTATIONS BY THE SAME DEVELOPER

	SIZING MODEL		COST MODEL
1.	ASSET-R	------------->	SOFTCOST-R SOFTCOST-ADA
2.	CEIS	------------->	SYSTEM-3
3.	PRICE SZ	------------->	PRICE S
4.	QSM SIZE PLANNER	------------->	SLIM
5.	SPQR SIZER/FP*	------------->	SPQR/20 SPQR/30
6.	BYL**	------------->	COCOMO

* In addition to being a stand-alone package, SIZER/FP is implemented within the SPQR/20 and SPQR/30 packages.

**BYL is a single package that includes both sizing and costing (COCOMO) capabilities.

3.2.5 Life-Cycle Phase for Application

Figure 3-1 shows the life-cycle phase at which each type of model is applied. Earlier estimates are characterized by greater uncertainty. As systems mature and processing functions are more clearly defined, the amount of uncertainty decreases. Uncertainty is further reduced by the application of risk management techniques whose primary purpose is the reduction of technical, schedule, and cost risks associated with software acquisition [REIF86, AFSC87].

Correlating Figure 3-1 with the models in this evaluation, the analogy approaches (SSA, ESD, and QSM Fuzzy Logic) can be applied earliest in the life-cycle during software requirements analysis. CEIS and SSM can also be applied in this phase. CEIS inputs require the user to analogize between project attributes of previous efforts and attributes of the developing

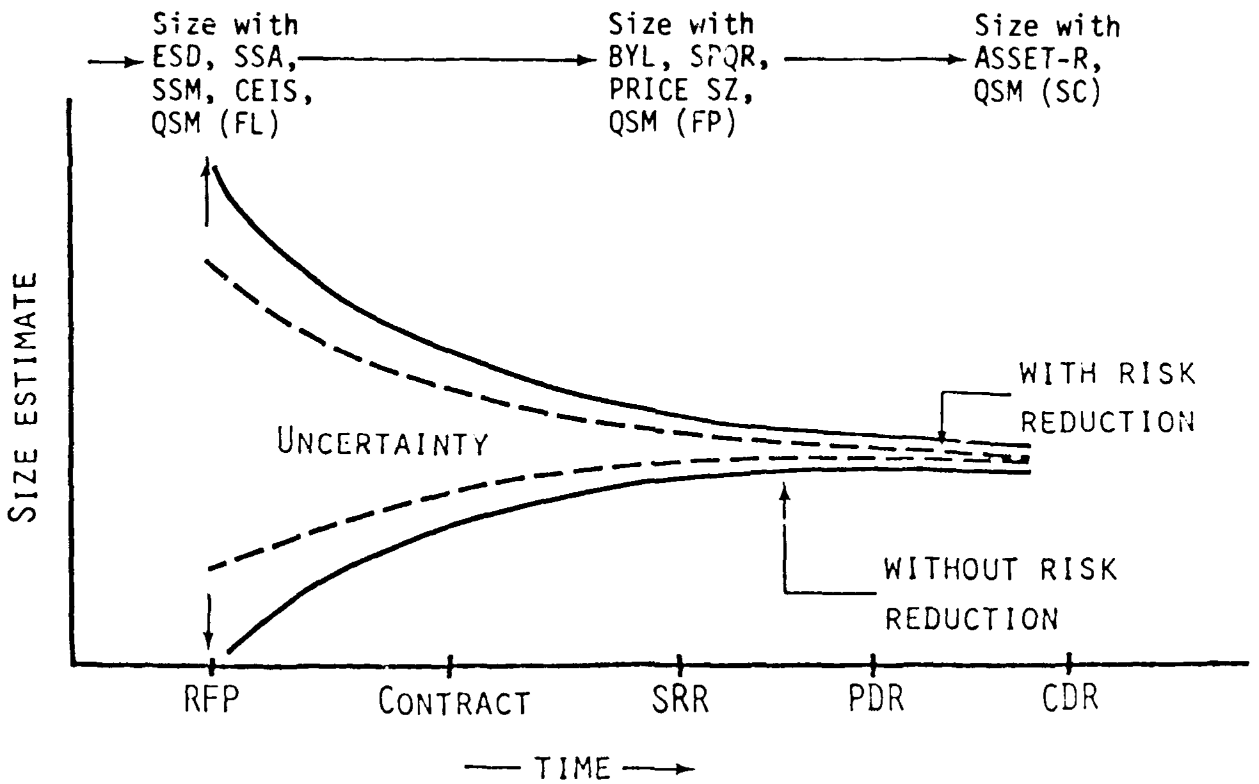


Figure 3-1. Use Different Approaches at Different Times During the Life-Cycle

system. Analysts who provide input to the SSM model are also, in effect, drawing on their own knowledge of similar, past efforts.

Function point techniques (SIZER/FP, BYL, and QSM Function Points) can be applied once external system design is complete at the preliminary design phase. The analyst applying the technique must have knowledge of how input screens and output reports/displays are to look. The data to be used by the application, as well as a general concept of how output is to be generated should be known. PRICE SZ is also applicable at the completion of this phase.

ASSET-R (linguistic approach), which requires definition of engineering formula equations in order to derive an operator and operand count, can be applied during the detailed design phase. Information required by the analyst for ASSET-R is the same as for function point techniques, except that definitive formulations to be used for internal processing and output generation are required.

QSM Standard Component Sizing is also applied in the detailed design phase. Standard Component Sizing is based upon the number of components the system will contain. The analyst's inputs to the model infer a knowledge of internal design of computer software components. Choosing the best method for software size estimation at a particular point in the life cycle must be done in accordance with the type of information available to the analyst.

3.2.6 Input Parameters

Appendix A contains a list of the inputs required by the nine models. In analyzing the models from the perspective of the input parameters required, it is necessary to view them from two different perspectives:

1. Discrete user inputs each model requires

2. The type/variety of information encompassed by model parameters.

The two perspectives provide an indicator of the amount of effort or tedium associated with applying the models. The following examples demonstrate how discrete inputs and input types are counted:

- Reference task size for three completed tasks is one (1) input type; three (3) discrete inputs.

- PERT sizing data (lowest possible size, most likely size, highest possible size) required for each system function is one (1) input type; three (3) discrete inputs per function.

- The number of system external inputs, rated to three levels of complexity (simple, average, and high) is one (1) input type; three (3) discrete inputs.

In order to look at the types of information encompassed by model parameters, they were classified into the following five categories:

- Qualitative inputs
- Quantitative inputs
- Identification inputs
- Modular (functional) components
- Calibration factors.

Counts associated with modular (functional) components are given on a per function basis.

Table 3-6 lists the number of discrete user inputs required for each model. The QSM Size Planner is treated as three distinct models since different input variables are required depending upon which of the three sizing techniques is to be implemented. The number of inputs range from 64 for ASSET-R to two (2) for SSA.

Table 3-7 provides the number of types of input required for each model. Figure 3-2 shows the distribution of parameter types across the categories. Over 43% of the input types are qualitative.

TABLE 3-6

DISCRETE INPUT PARAMETER COUNTS

SOFTWARE SIZING MODELS

CATEGORY	ASSET -R	BYL	CEIS	ESD	PRICE SZ	QSM FL	QSM FP	QSM SC	SIZER /FP	SSA	SSM
Qualitative	15	14	51		10	3	1	3	3		
Quantitative	9	17	3		10		25	12	5		
Identification	2	2	10		1	1	1	1	6		3
Modular				7						2	*10
Calibration		1			3				1		
TOTAL	26	34	64	7	24	4	27	16	15	2	13

* Note: Number of discreet inputs for SSM increases geometrically with the number of modules; that is for n modules, number of discreet inputs = n/2 * (n-1) + 5n.

TABLE 3-7

NUMBER OF INPUT TYPES PER MODEL

SOFTWARE SIZING MODELS

CATEGORY	ASSET -R	BYL	CEIS	ESD	PRICE SZ	QSM FL	QSM FP	QSM SC	SIZER /FP	SSA	SSM
Qualitative	15	14	3		10	3	1	3	3		
Quantitative	9	7	1		10		5	12	5		
Identification	2	2	3		1	1	1	1	6		3
Modular				7						2	8
Calibration		1			3				1		
TOTAL	26	24	7	7	24	4	7	16	15	2	11

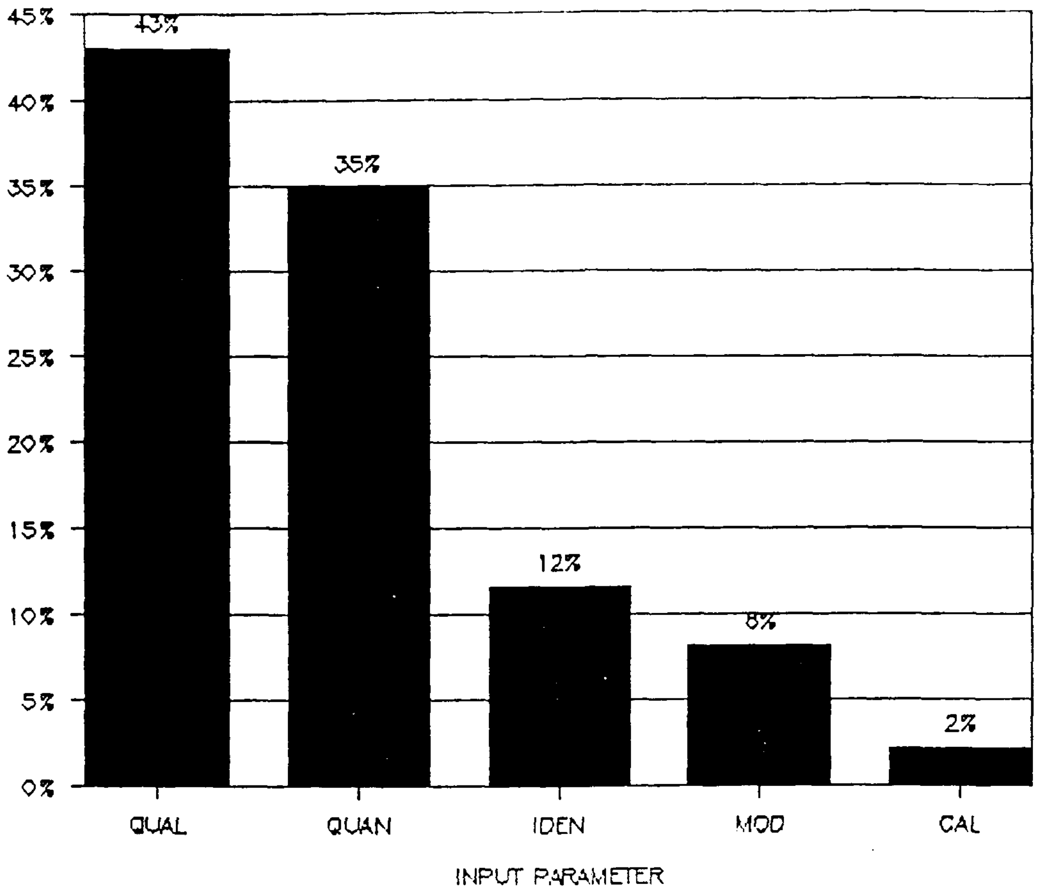


Figure 3-2. Distribution of Parameter Types by Input Categories.

3.2.7 Model Assessment

This section presents a set of factors that provide a basis for assessing each model. Each criterion is presented as a series of questions. Rationale for responses to these questions are in the form of detailed textual descriptions which are contained within the text that accompany each chart. The models are rated on a scale of 0 to 4 with a 0 indicating a complete absence of the stated criterion and a 4 indicating a high degree of the stated characteristic. Assessments are based upon actual use of the models or demonstrations, in addition to briefings and references provided by the model developers.

Assessment criteria are subdivided into the following subject categories:

- User input
- Historical data and analysis
- Underlying methodology
- Model output
- Model usability.

3.2.7.1 Assessment: User Input

The first category addresses user input to the sizing model. Inputs determine when the model can be applied and how much effort, training, and skill are required to obtain an estimate. Table 3-8 rates each model on the basis of its user input. An explanation of model ratings for each assessment criteria follows.

1. ARE THERE FEW INPUTS TO DERIVE?

Models are rated according to the number of discrete user inputs each requires to estimate the size for a new development effort. Identification inputs and other user inputs that have no direct affect on size are not considered in the rating. For ESD, SSA, and SSM, the number of inputs will depend upon the number of system functions or modules to be sized. The number of discrete user inputs required for each model are provided in Section 3.2.6, Table 3-6. Models are given a rating of 4 if the number of discrete inputs they require is less than five (5). Other ratings were assigned as follows:

- 6 to 10 inputs: rating = 3
- 11 to 15 inputs: rating = 2
- 16 to 20 inputs: rating = 1
- More than 20 inputs: rating = 0.

2. CAN THE MODEL BE APPLIED EFFECTIVELY WITHOUT KNOWLEDGE/EXPERIENCE IN THE APPLICATION AREA?

Four models (SSM, ESD, SSA, and CEI) require the user draw upon previous experience and familiarity with past projects. The basis of the estimate for SSM (rating = 0) is the user's personal experience whereas ESD, SSA, and CEIS estimates are facilitated by existing project data (ratings = 1).

Function point implementations (ASSET-R, BYL, QSM Function Points, and SPQR) and PRICE SZ (ratings = 3) are about the same relative to the extent of previous application experience recommended for their use. Initially, experience with similar applications is not required. However, estimates will generally improve with successive use of each model due to increased understanding of how they are applied.

TABLE 3-8

ASSESSMENT CRITERIA FOR USER INPUT

ASSESSMENT CRITERIA	MODEL										
	ASS	BYL	CEI	ESD	PRI	QSM FL	QSM FP	QSM SC	SPQ	SSA	SSM
1. ARE THERE FEW INPUTS TO DERIVE?	○	○	○	●	○	●	○	◑	◕	●	○
2. CAN THE MODEL BE APPLIED EFFECTIVELY WITHOUT KNOWLEDGE/EXPERIENCE IN THE APPLICATION AREA?	◕	◕	◔	◔	◕	●	◕	◑	◕	◔	○
3. ARE INPUTS EASILY UNDERSTOOD?	◑	◑	●	●	◔	●	◑	◕	◑	●	●
4. IS INPUT DATA AVAILABLE EARLY IN THE LIFECYCLE?	◑	◕	●	●	◕	●	◕	◑	◕	●	●

LEGEND				
●	◕	◑	◔	○
FULL CAPABILITY →				MINIMAL CAPABILITY

QSM Standard Compoents Sizing (rating = 2) is facilitated by previous application experience more than function point models but not to the extent of analogy approaches.

Fuzzy Logic (rating = 4) estimation is not affected by previous application experience.

3. ARE INPUTS EASILY UNDERSTOOD?

This criteria assesses the ambiguity or amount of interpretation associated with model input. Some of the model will require additional training or instruction to understand how to quantify or rate elements of a software system.

PRICE SZ (rating = 1) was rated as requiring the most training application. ASSET-R, BYL, QSM Function Points, and SPQR were each assigned a rating of 2 because users who are unfamiliar with the function point approach may require some training.

In comparison, input parameters for QSM Standard Components Sizing (rating = 3), CEIS, ESD, QSM Fuzzy Logic, SSA, and SSM are straitforward and require little further explanation.

4. IS INPUT DATA AVAILABLE EARLY IN THE SOFTWARE LIFECYCLE?

Analogy approaches (ESD, SSA and QSM Fuzzy Logic), SSM, and CEIS are applicable during requirements analysis (ratings = 4). BYL, SPQR, QSM Function Points, and PRICE SZ may be applied at the end of the preliminary design phase once external system characteristics (i.e. input screens, output reports/graphic displays) are known (ratings = 3). ASSET-R and QSM Standard Components Sizing whose input parameters require knowledge at the program implementation level are applicable at the end of the detailed design phase (ratings = 2).

3.2.7.2 Assessment: Historical Data and Analysis

Sizing is an evolving process where historical sizing data is used to develop and validate new methodologies, calibrate existing models, and derive new estimates on the basis of previous ones. Table 3-9 rates each model according to its ability to maintain and analyze a database of inputs, resulting estimates, and actual values.

1. IS THE SIZE ESTIMATE BASED UPON A DATABASE OF PREVIOUS ESTIMATES?

The SSM (rating = 0) is purely statistical in nature and does not base its current estimate on previous size estimates. At the other extreme, are database dependent models, ESD and SSA, which consist of a size database and an interface for selecting entries on which to base new estimates. Other database dependent models include CEIS which requires three completed tasks similar in nature to the task being estimated on which to base a new estimate. QSM Standard Components Sizing uses either the QSM historic database or the user's own historic data on which to base an estimate. These models were given a 4 rating.

Underlying equations for ASSET-R, BYL, PRICE SZ, QSM Fuzzy Logic, QSM Function Points, and SPQR were developed based upon analysis of project data. Each of these parametric models was assigned a rating of 2.

2. DOES THE MODEL ACTIVELY USE EARLIER USER DATA FOR SUBSEQUENT SIZE ESTIMATES?

Of the nine models in the evaluation, three are affected by a change or addition to the user's own historical data: CEIS, QSM Standard Components Sizing and SSA (ratings = 4). Calibration is more limited for BYL and PRICE SZ (ratings = 2). BYL will calculate and save a language expansion factor given the size, function point parameters, and processing characteristics of a completed effort. PRICE SZ will determine the Size Calibration Factor which is indicative of the user's organization expertise. Other models currently do not accept user data.

TABLE 3-9

ASSESSMENT CRITERIA FOR HISTORICAL DATA AND ANALYSIS

ASSESSMENT CRITERIA	MODEL										
	ASS	BYL	CEI	ESD	PRI	QSM FL	QSM FP	QSM SC	SPQ	SSA	SSM
1. IS THE SIZE ESTIMATE BASED UPON A DATABASE OF PREVIOUS EFFORTS?	◐	◐	●	●	◐	◐	◐	●	◐	●	○
2. DOES THE MODEL ACTIVELY USE EARLIER USER DATA FOR SUBSEQUENT SIZE ESTIMATES?	○	◐	●	○	◐	○	○	●	○	●	○
3. DOES THE MODEL MAINTAIN A DATABASE OF INPUTS, RESULTING ESTIMATES, AND ACTUAL VALUES?	◔	○	◔	○	○		◕		○	○	○
4. DOES THE MODEL PROVIDE SENSITIVITY ANALYSIS?	○	○	◔	○	○		●		○	○	○

LEGEND				
●	◕	◐	◔	○
FULL CAPABILITY	→			MINIMAL CAPABILITY

3. DOES THE MODEL MAINTAIN A DATABASE OF INPUTS, RESULTING ESTIMATES, AND ACTUAL VALUES?

The QSM Size Planner (Rating = 3) has a sophisticated historical data entry capability in which the user may record data for a new project or edit an existing project record. Project data for size, however, refers to actual size of a completed project and estimates by Size Planner are not automatically saved for later comparison to the actual size. CEIS (assigned a value of 1) will interface with the Mainstay Data Collection and Analysis System which retains sizes and all parameters for projects and contains metrics for sizing from extrapolation of the database. This is a separate product, however, that is available at an additional fee.

ASSET-R (rating = 1) interfaces with any spreadsheet package with DIF file format (Lotus, DBASE III, Framework, etc.). A spreadsheet package is also available from Reifer Consultants at no additional cost to the user. The package is not considered a part of the ASSET-R package.

All other models have no database maintenance capability.

4. DOES THE MODEL PROVIDE SUPPORT FOR SENSITIVITY ANALYSIS?

QSM Size Planner (rating = 4) will produce historical analysis reports for all systems contained in the current user's history database. Analysis support is alos provided through a CEIS interface to Main Stay Data Collection and Analysis System. CEIS was assigned a rating of 1 because the Mainstay interface is additional to the CEIS model. All other models do not provide support for sensitivity analysis.

3.2.7.3 Assessment: Underlying Methodology

The third assessment category shown in Table 3-10 addresses the models underlying methodology. Due to the newness of some of the models and lack of extensive testing in a variety of user environments, results of the test case study (described in Section 4) are the basis for rating the accuracy of several models.

1. IS THE MODEL A WHITE BOX?

Underlying theories of the BYL implementation of function points are in the public domain (rating = 4). Source code for the SSA and ESD models are also available on a more limited basis. Each of these models was assigned a rating of 4.

The underlying theories of CEIS (rating = 2) which is based upon some work by Dr. Joseph Lambert are public domain. CEI's implementation of the Lambert model, however, is proprietary.

SPQR unadjusted function point calculation is public domain. SPQR was assigned a 2 rating however because adjustment for processing complexity is internal to the model.

Each of the remaining models are "black box" and are accordingly rated at 0.

2. IS THE MODEL APPLICABLE TO DIFFERENT USER ENVIRONMENTS?

SSA (rating = 1) should be used primarily to size space systems software.

Function point approaches (BYL, QSM Function Points, and SPQR) have been validated for data processing environments. However, it has been suggested that these models will provide unsatisfactory results for scientific and real-time systems. To date, IITRI has not been able to attain access to published studies that address this finding. These models are currently assigned ratings of 2.

The ESD package (rating = 2) should be used to size DoD operational and support functions.

Remaining models are applicable in any user environment.

TABLE 3-10

ASSESSMENT CRITERIA FOR UNDERLYING METHODOLOGY

ASSESSMENT CRITERIA	MODEL										
	ASS	BYL	CEI	ESD	PRI	QSM FL	QSM FP	QSM SC	SPQ	SSA	SSM
1. IS THE MODEL A "WHITE BOX?"	○	●	◒	●	○	○	○	○	◒	●	○
2. IS THE MODEL APPLICABLE TO DIFFERENT USER ENVIRONMENTS?	●	◒	●	◒	●	●	◒	●	◒	◔	●
3. ARE EQUATIONS PARAMETRIC-BASED?	●	●	○	○	●	●	●	●	●	○	○
4. IN THE TEST CASE STUDY * DID THE MODEL PROVIDE ACCURATE RESULTS?	●	◒	NA	○	◒	NA			◔	NA	●

* The test case application is described in Section 4.

NA = Not Applied

LEGEND				
●	◕	◒	◔	○
FULL CAPABILITY →				MINIMAL CAPABILITY

3. ARE EQUATIONS PARAMETRIC-BASED?

Formulations used in ASSET-R, BYL, PRICE SZ, QSM Size Planner, and SPQR (ratings = 4) were developed, based upon an analysis of project data. CEIS, ESD, SSA, and SSM (ratings = 0) are purely statistical in nature.

4. IN THE TEST CASE STUDY, DID THE MODEL PROVIDE ACCURATE RESULTS?

Ratings for accuracy of applied models in the test case study (described in Section 4) are based upon resultant estimates as compared to actual size. The relative error determined for each model is provided in Table 3-11. Ratings from 0 to 4 coincide with the following ranges for relative error:

Relative error < 30%, rating = 4
31% - 100%, rating = 3
101% - 200%, rating = 2
201% - 300%, rating = 1
301% - 400%, rating = 0.

TABLE 3-11

RELATIVE ERROR DETERMINED FOR SELECTED MODELS IN TEST CASE APPLICATION

MODEL	RELATIVE ERROR
ESD	310%+
SPQR	291%
BYL	144%
PRICE SZ	133%
ASSET-R	30%
	28%
SSM	27%

3.2.7.4 Assessment: Model Output

Each model provides a source lines of code estimate. Some of the models, in addition, produce a measure of the functionality of the software as an alternate output. Table 3-12 rates each model according to its output reporting capabilities.

1. IS THERE A PROBABILITY ASSOCIATED WITH THE ESTIMATE?

Probability ranges are projected for all of the models except BYL, SSA, and SPQR. These three models are assigned 0 ratings.

2. DOES THE MODEL ESTIMATE FUNCTION POINTS?

ASSET-R, BYL, SPQR, and SSM (ratings = 4) will estimate function points. (The output of SSM is in the unit selected for the reference modules.) Remaining models are rated at 0.

3. ARE INPUTS SUMMARIZED ON THE OUTPUT REPORT?

User inputs are provided on all output summary reports with the exception of CEIS and SSM.

4. DOES THE MODEL PROVIDE GRAPHICS CAPABILITIES?

Three models provide graphics capabilities. QSM (rating = 4) provides a number of color displays and line graphs. ASSET-R (rating = 4) graphics are in the form of histograms. ESD (rating = 4) provides a probability curve of the size range at the confidence level prescribed by the user. SSA (rating = 2) graphics are character oriented.

TABLE 3-12

ASSESSMENT CRITERIA FOR MODEL OUTPUT

ASSESSMENT CRITERIA	MODEL								
	ASS	BYL	CEI	ESD	PRI	QSM	SPQ	SSA	SSM
1. IS THERE A PROBABILITY ASSOCIATED WITH THE ESTIMATE?	●	○	●	●	●	●	○	○	●
2. DOES THE MODEL PRODUCE A FUNCTION POINT ESTIMATE?	●	●	○	○	○	○	●	○	●
3. ARE INPUTS SUMMARIZED ON THE OUTPUT REPORTS?	●	●	○	●	●	●	●	●	○
4. DOES THE MODEL PROVIDE GRAPHICS CAPABILITIES?	●	○	○	●	○	●	○	◑	○

LEGEND

● ◕ ◑ ◔ ○

FULL CAPABILITY → MINIMAL CAPABILITY

3.2.7.5 Assessment: Model Usability

The final category presented in Table 3-13 rates each model according to its user-friendliness and vendor/developer user support.

1. DOES THE MODEL HAVE A USER-FRIENDLY INTERFACE?

Analysts rated the BYL (rating = 4) user interface as superior because every input or modification to parameters will result in the immediate calculation and display of a new size. ASSET-R (rating = 4) uses detailed help screens and worksheets to assist the analyst in quantifying the estimate. Other models (CEIS, QSM Size Planner, SPQR, and SSM) have good ratings (3) for user interface. ESD and SSA (ratings = 2) were given lower ratings because of the run time associated with their database search processes. PRICE SZ was rated at 2 due to inconveniences associated with time-sharing.

2. IS USER SUPPORT AVAILABLE FOR ASSISTANCE IN APPLYING THE TECHNIQUE?

ASSET-R, BYL, CEIS, PRICE SZ, QSM Size Planner, and SPQR are fully supported (rating = 4). The ESD, SSA, and SSM are supported on a more limited basis (rating = 2).

TABLE 3-13

ASSESSMENT CRITERIA FOR MODEL USABILITY

ASSESSMENT CRITERIA	MODEL								
	ASS	BYL	CEI	ESD	PRI	QSM	SPQ	SSA	SSM
1. DOES THE MODEL HAVE A USER-FRIENDLY INTERFACE?	●	●	◕	◒	◒	◕	◕	◒	◕
2. IS USER SUPPORT AVAILABLE FOR ASSISTANCE IN APPLYING THE TECHNIQUE?	●	●	●	◒	●	●	●	◒	◒

LEGEND				
●	◕	◒	◔	○
FULL CAPABILITY	→	→	→	MINIMAL CAPABILITY

4.0 TEST CASE STUDY: APPLICATION OF SELECTED SIZING MODELS TO AN EXISTING SYSTEM

4.1 BACKGROUND

To gain additional insight into the strengths and weaknesses of each model, IITRI project personnel applied these available models to an existing Air Force system:

- ESD
- BYL
- ASSET-R
- SSM
- SPQR
- PRICE SZ

SSA, which was also available to project personnel, was not applied. SSA was developed primarily for use in sizing space system applications; it is not applicable to the Air Force system which was the subject for the sizing exercise.

The exercise focused on the derivation of model inputs and on the accuracy of resulting estimates. These aspects of sizing - inputs and accuracy - are discriminating factors for determining which model is most appropriate for an intended use. Inputs determine when a model can be applied. They will vary considerably, depending upon the particular model. Some models require more definitive information about a software job. They require more effort to apply. However, they generally provide more accurate size estimates.

Other models are designed to give only "ballpark" estimates when there is a lack of definitive information about a software job. Understandably, these models requre few inputs but they draw upon the knowledge and experience of the estimator.

The following sections are provided to convey what is involved in deriving a size estimate utilizing various sizing approaches. Difficulties encountered are explained and recommendations are made to overcome some of these. In addition, the training required to utilize a particular sizing model is indicated as well as the need of accessibility to those project personnel who developed the software. Organization of the test case study is as follows:

- Section 4.2 is an overview of the Air Force system that is the subject for the sizing exercise.

- Section 4.3 describes application of the analogy approach using the ESD Software Sizing Package.

- Section 4.4 discusses an SSM application to derive a size estimate. Since PERT sizing data was required input to the SSM model, size estimates were also derived using the PERT methodology. PERT estimates are provided for comparison with the SSM output.

- Section 4.5 focuses on the derivation of function point parameters for the select Air Force system. Function point counts are obtained using the approach implemented by the BYL model.

- Section 4.6 discusses function point estimates derived using the approach implemented by SIZER/FP.

- Section 4.7 addresses the derivation of operator and operand counts for the ASSET-R model. These parameters are used in an application of the ASSET-R model to the subject Air Force system.

- Section 4.8 reviews a sizing exercise utilizing PRICE SZ.

4.2 OVERVIEW OF THE CATSS SENSITIVITY MODEL

The subject for the sizing model exercise is an Air Force system called Cartographic Applications for Tactical and Strategic Systems (CATSS) Sensitivity Model. The CATSS Sensitivity Model is a laboratory tool, developed by IIT Research Institute and Hughes Aircraft Company, that demonstrates the operation of select Air Force weapons systems relative to their use of cartographic data bases. It provides a quantitative measure of system performance and demonstrates the effects on system performance when various data base parameters are degraded (i.e., accuracy, resolution, feature content, feature attributes, etc.).

4.2.1 System Organization

The CATSS Sensitivity Model is composed of the following five distinct subsystems, each one an independent model of a function used in either tactical or strategic systems [CATS86a]:

1. Navigation Function (Nav)
2. Threat Avoidance Function (TA)
3. Reconnaissance/Surveillance Function (R/S)
4. Cross-Country Movement Function (CCM)
5. Cockpit Display Function (CD)

Functions are executed from three separate programs. The main program contains all instruction for execution of the Nav, TA, and R/S functions. In addition, this main program calls VAX system-level FORTRAN routines to spawn sub-processes which run the separate CCM and CD programs.

Input to the system consists of cartographic data sets, which must be ready for use before the system is run. Terrain data which is accessed through execution of any of the functions should be considered to be in the proper format for retrieval. (Source code required to load terrain data into the relational database is not considered in the sizing exercise).

Other parameters required for execution of the system are entered by the user at run time in response to queries by the system or by direct entry in screen menus. Modelling response time depends upon the function being processed and on the amount of cartographic data being processed.

Output from the system includes screen summaries of analysis results, report files for hard copies of run results, and various graphic displays.

4.2.2 User Interface

Processing and output is dependent upon the function selected for analysis. When the analyst invokes the CATSS Sensitivity Model, the Main Menu is displayed. A function is selected by positioning the cursor (using the arrow keys or mouse) in front of the desired option and entering a carriage return. Each option results in the main menu of the selected subsystem being displayed on the screen. Figure 4-1 is a top level functional flow diagram that shows how each subsystem menu is accessed from the Main Menu.

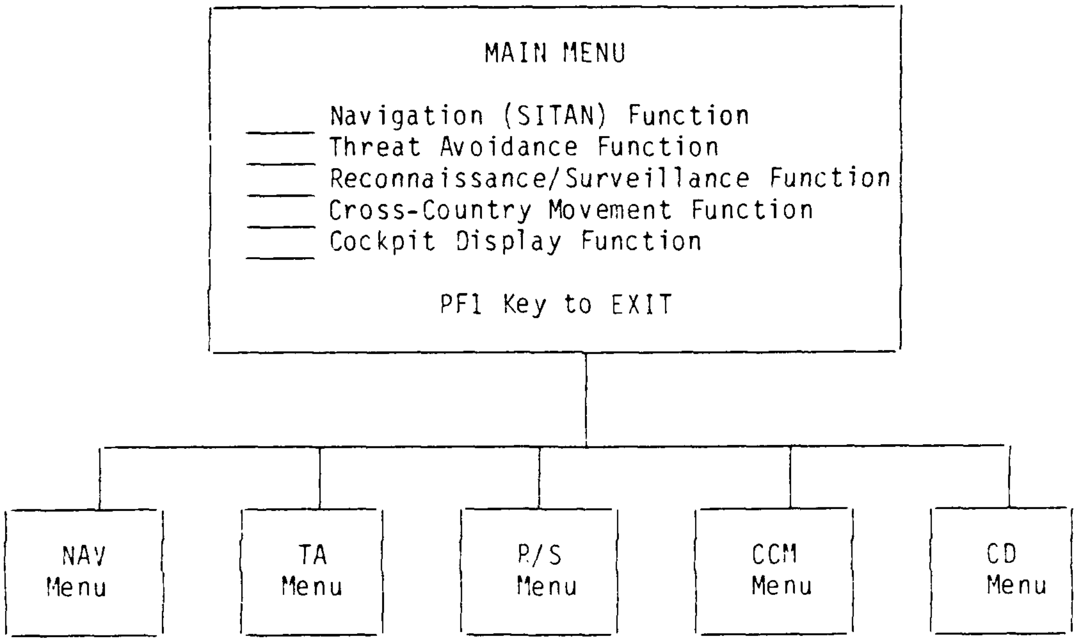


Figure 4-1. Top-Level Functional Flow from the Main Menu to each Subsystem Menu.

Each subsystem level menu is a selection menu that indicates a specific procedure that the user may wish to perform, depending upon the selected subsystem:

- define degradation parameters and create a degraded database,
- specify subsystems parameters,

- designate source data for analysis,
- choose a flight path,
- display a report, etc.

Figures 4-2 and 4-3 show functional flow for the Navigation function and Threat Avoidance function, respectively. Options are selected by positioning the cursor (using arrow keys) in front of the desired option and entering a carriage return.

For the Navigation function, selection of one of the first two options results in the display of an input screen which is indicated by the arrows in Figure 4-2. The input screens show parameters with default values in parenthesis. The user may scroll through the data items on the screen; changing default values by typing at the current cursor position. The selection of the third Navigation option, Choose Path, results in the display of a terrain contour map within which the path is drawn on the screen via cursor. When all parameters have been specified, the fourth option is used to initiate a Navigation run and generate summary statistics of the analysis. Appendix D illustrates some of the model generated output. The PF1 key returns control back to the Main Menu.

The Threat Avoidance function (Figure 4-3) and remaining functions are executed in a similar manner.

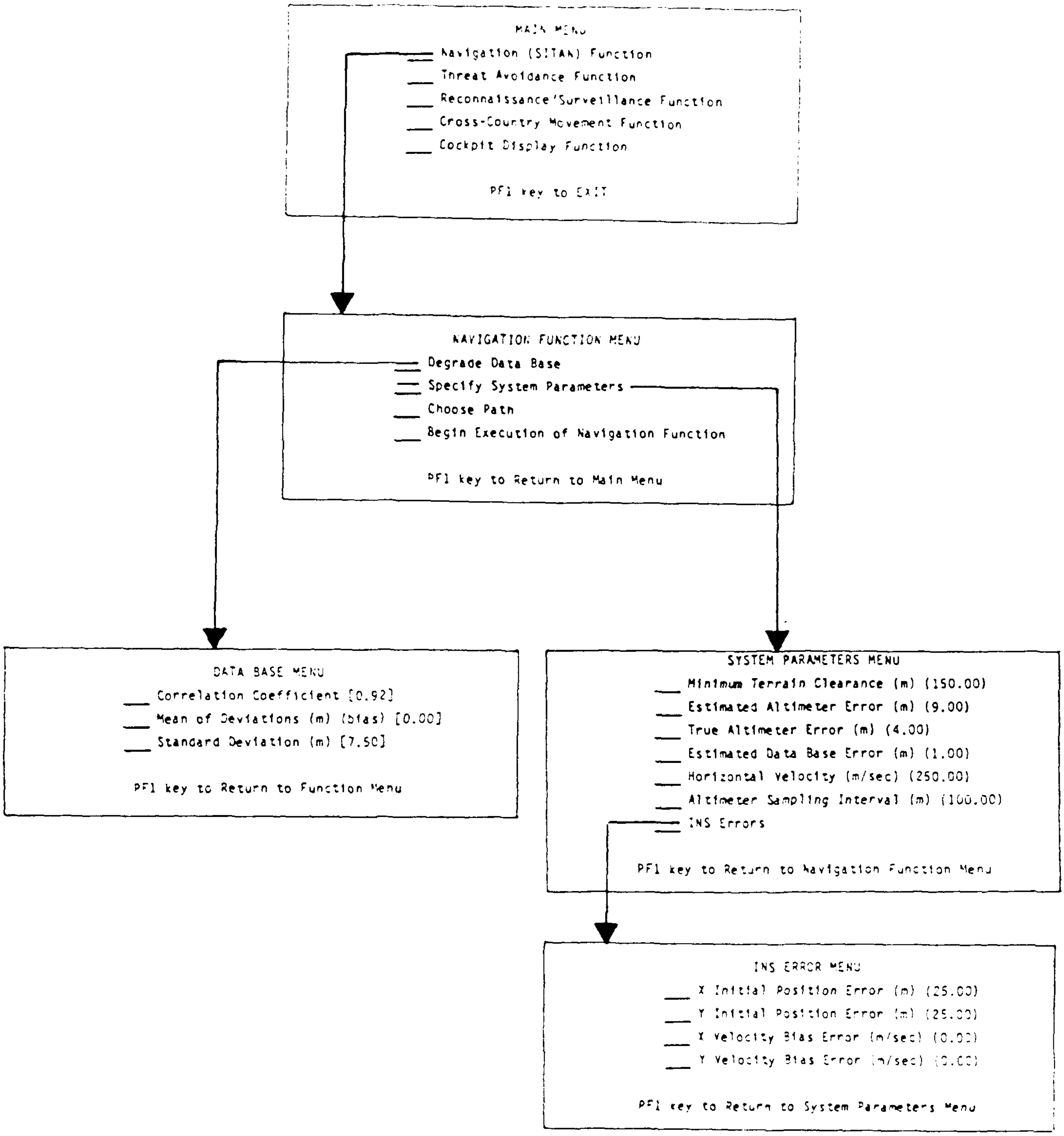


Figure 4-2. Functional Flow for the Navigation Subsystem.

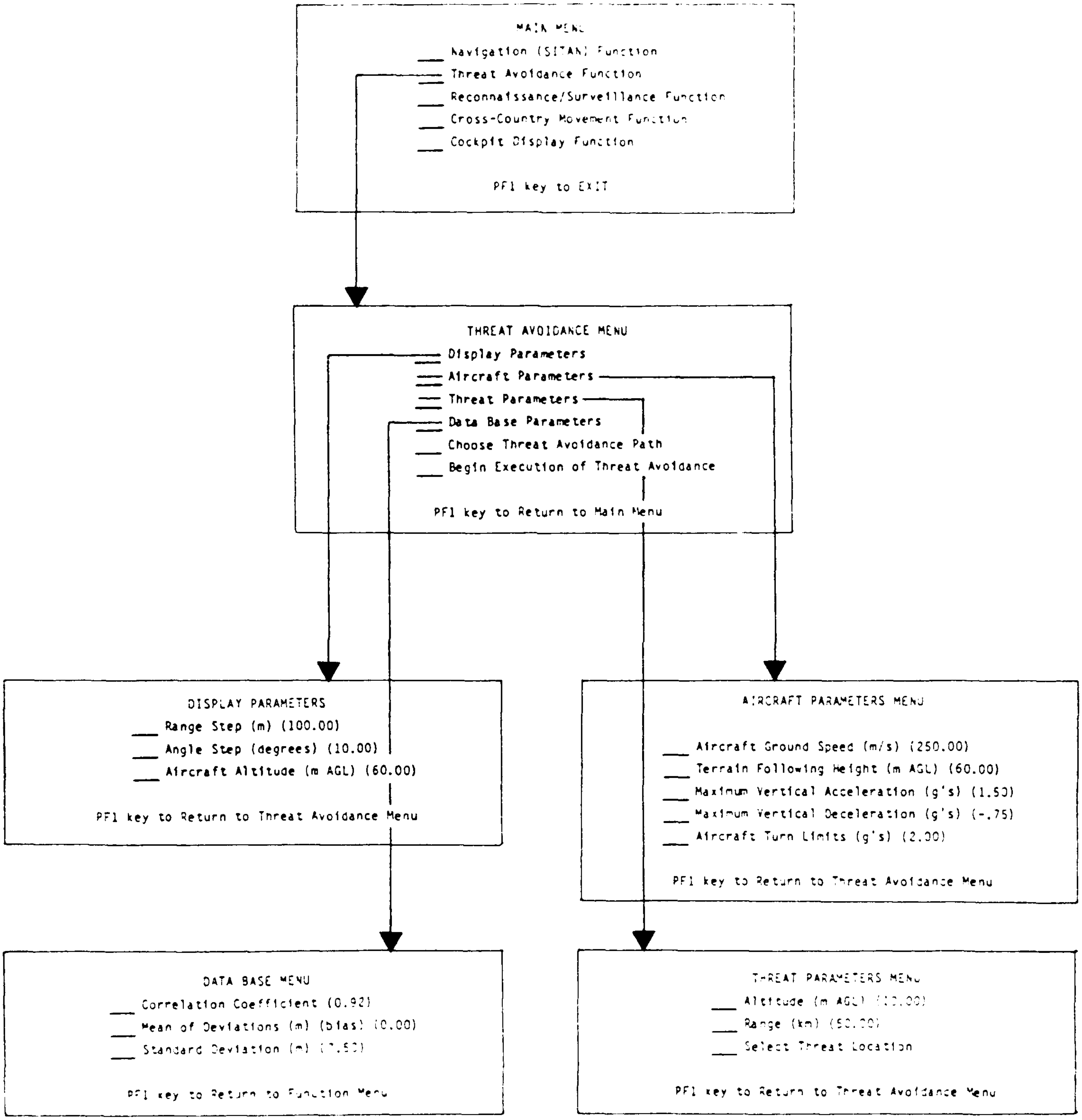


Figure 4-3. Functional Flow for the Threat Avoidance Subsystem.

4.3 APPLICATION OF THE ESD SIZING PACKAGE

The underlying objective in implementing this approach is to correlate software function categories listed for the ESD database (See Table A-3 in Appendix A) with CATSS functions. This suggests that the analyst must decide the level at which to define CATSS subsystem components for correlation with ESD index numbers.

4.3.1 Functional Decomposition of CATSS Software

The initial inclination was to assign ESD index numbers at the subsystem level. That is, to assign an index number to each of the Nav, TA, R/S, CCM, and CD functions. Subsequent analysis, however, showed that this would be misleading because CATSS functions are too specialized. For example, the CATSS Navigation subsystem function correlates to ESD index number 2.2, Navigation. However, the CATSS Navigation function which demonstrates how the ability to track the position of an aircraft is affected by inaccuracies in the stored elevation database, does not reflect typical navigation implementation in the sense in which it is defined by ESD. Likewise, the CATSS Cross-Country Movement function, modeled to demonstrate the sensitivity of CCM analysis to the resolution of input feature data, should not be equated to Mission Planning index 2.1. To further cloud the issue, each of the five subsystems is a performance simulation which is index 12.5. Each offers computer operator interface (index 4.2) and each implements on-line database retrieval and output (index 5.1). Thus, at this level, it was difficult to correlate CATSS subsystems with a specific ESD index.

The approach that was taken to define CATSS functional components was to treat each subsystem as an independent entity. Components of each subsystem were reviewed and it was noted that there was a great deal of overlap between subsystem functions. A list of the unique CATSS functions was compiled from all subsystem functions.

The functional components of the CATSS Sensitivity Model were initially defined by an analyst who reviewed the CATSS Sensitivity Model Functional Description [CATS86a] and Users Manual [CATS86b]; and in addition executed the CATSS Sensitivity Analysis software. The initial list of software functions derived by the analyst was reviewed by the Project Manager for the CATSS

effort. He provided additional input which led the analyst to revise the modular breakdown to that in Table 4-1.

Functional decomposition is not always clear-cut. Because of its comprehensive nature, it is time consuming although this will vary depending upon the nature and complexity of the system. The process is best performed by someone familiar with the design or development of the software or similar application.

It should also be emphasized that this exercise was performed after the product was developed. The analyst had the benefit of project documentation.

TABLE 4-1

FUNCTIONS OF THE CATSS SENSITIVITY MODEL

1. Executive
2. Elevation degradation
3. Feature degradation: Areal, linear, and point
4. Flight path definition
5. Demonstration and initialization of parameters (User Interface)
6. SITAN Navigation
7. ADLAT terrain-following algorithm
8. Intervisibility computation
9. CCM algorithm
10. Data retrieval
11. Measure of performance: true vs. degraded
12. Output files/screen displays

4.3.2 Correlation of ESD Indexes to CATSS Functional Components

Once defined, the next step was to assign an index number(s) shown in Table A-3 (page A-14) to each CATSS module. In some cases there was uncertainty that a particular module fit an ESD category. In discussing the situation with ESD personnel, it was decided to send them a description of CATSS modules for assignment of index numbers and application of the ESD sizing approach.

Table 4-2 summarizes the indexes assigned by Captain Joe Dean and Barbara Mentzel at ESD. The functional characteristics for modules 2, 3, and 11 could not be determined. For these modules, a knowledgable engineer should use

TABLE 4-2

CATSS FUNCTIONS CORRELATED TO ESD FUNCTIONS

	MODULE NAME	INDEX NUMBER	ESD FUNCTION CATEGORY
1.	Executive	4.1/4.4	Computer Resource Management/ Special Device Interface
2.	Elevation degradation	?	
3.	Feature degradation	?	
4.	Flight path definition	2.4	Sighting, Designation, and Location Determination
5.	User interface	4.2	Computer Operator Interface
6.	SITAN Navigation	2.2	Navigation
7.	ADLAT terrain-following algorithm	2.3	Aircraft Steering and Flight Control
8.	Intervisibility computation	2.4	Sighting, Designation, and Location Determination
9.	CCM algorithm	2.1	Mission Planning
10.	Data retrieval	3.8	Data Reduction
11.	Measure of performance	?	
12.	Output files/screen displays	4.2	Computer Operator Interface

discretion combining other selection criteria in order to obtain the data on which to base an estimate.

Once indexes were determined for the CATSS Sensitivity Model, Barbara Mentzel applied the ESD package and provided an overview of the process [MENT87]. The sizing package was run once for each of the nine modules, taking a total of about 2-3 hours. The most time-consuming aspect of the package is the Condor database search process, during which time the user is free to return to his/her desk. The user is alerted at the point additional interaction is required.

TABLE 4-3

ESD OUTPUT FOR THE CATSS SENSITIVITY MODEL

Module Number	INDEX NUMBER	# of Data Points	Data Range in SLOC	80% Confidence	50% Confidence	Most Likely
1.	4.1/4.4	47	100 - 32,300	1 - 23,123	1 - 16,646	5,500
2.	?					?
3.	?					?
4.	2.4	13	152 - 132,500	1 - 8,519	1 - 6,545	3,150
5.	4.2	39	12 - 10,991	1 - 5,951	1 - 4,242	4,242
6.	2.2	13	18 - 5,000	1 - 4,101	1 - 2,961	1,000
7.	2.3	14	1,037 - 16,000	1 - 15,838	1 - 12,259	6,100
8.	2.4	13	152 - 132,500	1 - 8,519	1 - 6,545	3,150
9.	2.1	6	252 - 50,032	1 - 55,246	1 - 39,921	13,550
10.	3.8	5	193 - 9,195	1 - 11,001	1 - 7,895	2,550
11.	?					?
12.	4.2	39	12 - 10,991	1 - 5,951	1 - 4,242	1,300
SUMMARY:		5 - 47	12 - 132,500	9 - 138,249	9 - 101,256	37,600

4.3.3. Results and Conclusions

Table 4-3 provides ESD estimates for the CATSS Sensitivity Model. The expected size of 37,600 SLOC does not include estimated sizes for modules 2, 3, and 11. The table shows the size of module 1 was determined by selecting entries in the ESD database with assigned indexes of either 4.1 (Computer Resource Management) or 4.4 (Special Device Interface). A total of 47 entries were retrieved via the automated database search process. Sizes ranged from 100 to 32,300 SLOC. Statistical analysis yielded a most likely size of 5,500 SLOC.

An actual copy of the final output was not available for inclusion in this report. However, it would be similar to that in Table 4-3 with the following two exceptions:

1. Data would be given for only a single module, and

2. Only one user determined confidence range would be displayed.

A breakdown of CATSS size on a module by module basis is available only for modules 5 and 9. Sizes for modules were obtained by an analyst who reviewed the CATSS Sensitivity Model Program Maintenance Manual [CATS86c]. The manual describes each of the 175 FORTRAN programs, subroutines, and functions which are used by the CATSS system. Routines associated with selected functions (i.e., user interface and the CCM algorithm) were designated by the analyst and subsequently counted using an automated tool.

Table 4-4 compares the actual versus ESD estimated size for each of these modules. The results, in conjunction with the summary output in Table 4-3 reveal estimates that are larger than the actual size.

The extent to which estimates departed from the actual size would have been much less, had the analyst implementing the software been more familiar with the CATSS application area. For example, in Table 4-3 the size range for module 9, the CCM algorithm, is 252 to 50,032 SLOC. The CCM algorithm is essentially speed computation for a specified vehicle within an area of known natural features (i.e., slope, soil type, surface roughness, season of the year, etc.). Intuitively, personnel familiar with CCM through related efforts, "know" that this function would be on the smaller side of the specified range. Had the large size entries been discarded, prior to analysis, an estimate closer to the actual, would have been obtained.

TABLE 4-4

ACTUAL VERSUS ESD ESTIMATED SIZE FOR THE CATSS SENSITIVITY MODEL

	ESD SLOC	ACTUAL SLOC
USER INTERFACE (MODULE 5)	4,242	2,300
CCM ALGORITHM (MODULE 9)	13,550	650

4.4 APPLICATION OF THE SSM

There are currently two versions of the SSM. The time-share version developed in 1981 is accessible only to commercial users. The PC version released in September 1987 is available to both commercial users and to the DoD. The differences in the two versions are reflected in the basis of the estimates; the same type of information but not as much information is input to the PC version. Specifically, dissimilarities in the two versions are as follows:

- The time-share version requires a minimum of three analysts to provide sizing input. The PC version requires sizing input from one analyst.
- The maximum number of modules that SSM will size is 33 in the time-share version. The upper limit to the number of modules in the PC version is constrained by the availability of memory of the computer. (For a 512K configuration, the maximum number of modules is 300.)
- The modules must include two modules of known size in the time-share version. A minimum of two modules must be of known size in the PC version.

Though this report addresses the PC version of the model, the commercial time-share version is the one to which IITRI personnel had access for application to the CATSS Sensitivity Model. The differences in the two versions will have most impact on the standard deviation of the resulting estimates. Statistical analysis performed to obtain the resultant estimate is the same in both versions.

4.4.1 Procedure

Application of the SSM is both an interactive and manual process. The first step requires the analyst to describe project characteristics to the model. Information input to the SSM for the CATSS Sensitivity Model is summarized in Table 4-5. The data was read into an initial input file that was later used to generate input data forms.

TABLE 4-5

SSM INITIAL INPUTS FOR THE CATSS SENSITIVITY MODEL

Company/Organization: IIT Research Institute

Project Name: CATSS

Coordinator: K. Slack

Maryland Technology Center
4550 Forbes Boulevard, Suite 300
Lanham, Maryland 20706-4324
(301) 459-3711

Number of Modules: 12

Module Names:

1. Executive
2. Elevation degradation
3. Feature degradation
4. Flight path definition
5. User Interface
6. SITAN Navigation
7. ADLAT terrain-following algorithm
8. Intervisibility computation
9. CCM algorithm
10. Data retrieval
11. Measure of performance
12. Output files/screen displays

Range of Module Sizes: 200 SLOC - 5000 SLOC

Data forms were distributed to four programmer/analysts whose years of software development experience were 5, 7, 10, and 21 years respectively. Each had worked on similar efforts. Only one had worked on the CATSS effort in the status of Project Manager. None of the four analysts had a preconceived notion as to the size of the CATSS project.

The coordinator instructed the analysts to treat each set of data forms separately. That is, complete one set of relative ranking data, set it aside and forget about it, then go on to fill out the next set. It did not matter if the analyst was inconsistent between the four data set forms.

In addition to receiving the data set forms, each participant in the sizing exercise was cited two modules of known size: the user interface

module (5) consisted of 2300 lines of executable FORTRAN source code and the CCM algorithm (9) was 650 lines. Reference module sizes were derived by the coordinator who determined which software routines performed these functions from the Program Maintenance Manual [CATS86c]. An in-house tool was used to count the lines of code for designated modules.

Also provided to each analyst was a brief written explanation of each module. The following description for two of the CATSS modules illustrates the amount of detailed description provided to each analyst:

(2) Elevation degradation

> Altitudes are degraded by the addition of errors. Input parameters for the noise function are the standard deviation of the error and the correlation coefficient of the errors. Generally, an array of grid points is filled with random variables. The array is filtered in the x and y directions. The process yields an error value which is added to the elevation value at the corresponding grid point.

(7) ADLAT terrain-following algorithm

> An ADLAT flight trajectory consists of a series of parabolic arcs representing either a pull-up maneuver to clear a peak or a push-over maneuver to fly down a valley once a peak has been cleared. This module tests for clearance over terrain points and computes the parabola.

Each analyst was told to contact the coordinator if he wanted additional information about each module.

Upon receipt of the completed input data forms from the analysts, the coordinator input the information to the SSM model. For the coordinator, this was the most tedious part of the exercise. For each participant in the sizing exercise the following number of inputs were entered to the model:

Where n = number of modules, for

Pairwise comparison data: n/2 x (n-1)
Ranking data: n
Sorting data: n
PERT sizing data: 3 x n

Hence, for 12 modules, the coordinator entered a total of 126 values per analyst which were saved into system data files (for optional review or modification). Statistical analysis was performed on the data files when the SSM was executed in the final step of this process.

4.4.2 Results and Conclusion

Figure 4-4 provides SSM estimates for the CATSS Sensitivity Model. The expected size of 11,700 represents a relative error of 27 percent based on the actual size of 9,177 SLOC.

After the results were obtained the coordinator of the exercise called a meeting of the participating analysts to elicit comments and recommendations. The coordinator used the PERT sizing data input to the model and manually derived a size estimate for each of the analysts to provide some basis for comparison. A summary of PERT sizing versus SSM versus the actual size is provided in Table 4-6.

Comments provided to the coordinator were as follows:

- Participants in the sizing exercise should have met and discussed the modules before filling out the forms. Module descriptions were not clear enough that everyone was thinking in the same vein. This was evident in the descrepancy between individual SSM inputs.

- Two of the four analysts thought that the size range provided by the coordinator was too constraining. Each perceived some of the modules to be less than 200 SLOC.

- The analysts agreed that filling out the data forms was tedious; especially the pairwise comparison data set (66 comparisons).

TABLE 4-6

PERT and SSM SIZE ESTIMATES FOR THE CATSS SENSITIVITY MODEL

		SLOC	STANDARD DEVIATION
PERT	Analyst #1	16,801	597
	Analyst #2*	15,050	314
	Analyst #3	14,801	405
	Analyst #4	13,132	213
SSM		11,700	850
ACTUAL		9,177	

* Analyst #2 was the CATSS Project Manager

IIT RESEARCH INSTITUTE

CATSS

S S M   S I Z E   E S T I M A T E S

CODE NO.	MODULE NAME	-SIGMA	EXPECTED MODULE SIZE	+SIGMA	STD. DEVIA-TION
1	EXECUTIVE	139	180	221	41
2	ELEVATION DEGRADATION	330	450	570	120
3	FEATURE DEGRADATION	490	670	850	180
4	FLIGHT PATH DEFINITION	350	460	570	110
5	USER INTERFACE	2300	2300	2300	0
6	SITAN NAVIGATION	640	830	1020	190
7	ADLAT TERRAIN-FOLLOWING ALG.	500	690	880	190
8	INTERVISIBILITY COMPUTATION	610	820	1030	210
9	CCM ALGORITHM	650	650	650	0
10	DATA RETRIEVAL	1120	1500	1880	380
11	MEASURE OF PERFORMANCE	320	440	560	120
12	OUTPUT FILES/SCREEN DISPLAYS	2080	2700	3320	620

S O F T W A R E   S I Z E   S U M M A R Y

EXPECTED S/W SIZE: 11700

STANDARD DEVIATION: 850

S O F T W A R E   S I Z E   R A N G E S

THE ACTUAL S/W SIZE WILL BE IN THE RANGE

	FROM:	TO:
50% PROBABILITY:	11200	12300
68% PROBABILITY:	10900	12600
95% PROBABILITY:	10000	13400
99% PROBABILITY:	9200	14300

Figure 4-4. SSM Output for the CATSS Sensitivity Model.

4.5 APPLICATION OF THE BYL MODEL

The BYL implementation of function points is the same approach taught at the IBM Function Point Analysis Workshops. As described earlier the function point measure is accomplished in three general steps:

1. Identify, classify, and count the five function types.
2. Determine degree of influence of processing characteristics.
3. Calculate the function point total.

4.5.1. Identification and Classification of Function Point Parameters

Appendix D describes the process to identify FP parameters for the CATSS Sensitivity Model. The following points summarize a few observations relative to parameter identification:

- The function point approach can be applied at the time in which external software characteristics of the system are known. System inputs, output displays/reports and interfaces need to be quantified.
- The analyst applying the model must know what type of data is to be used by the application.
- The analyst should generally know how the application output is to be generated.

Whether this information is known early on depends upon the application. For CATSS, much of the external design was presented in the proposal that was the response to a statement of work. Often, requesting agencies require contractors to give their perception of how a system is to look and how it is to be implemented based on the requirements. Therefore, it is not inconceivable that function points could be applied according to a proposed design. The accuracy of the results will depend upon the extent to which developed system varies from the proposed design.

Appendix D does not illustrate how each CATSS function point parameter was categorically defined as simple, average, or complex. This was performed by considering the number of data element types and logical internal file types referenced for each parameter. Data element types can be regarded as unique fields. For example, the DATA BASE MENU in Figure 4-5 is an external input. Three data element types (fields) are entered to the screen: correlation coefficient, mean of deviations, and standard deviation. There is

DATA BASE MENU

___ Correlation Coefficient [0.92]

___ Mean of Deviations (m) (bias) [0.00]

___ Standard Deviation (m) [7.50]

PF1 key to Return to Function Menu

Figure 4-5. Three Data Element Types are Entered to the Data Base Degradation Menu. One Logical Internal File Type is Referenced to Generate the Menu Display.

only one logical internal file type referenced to generate the menu display. The number of file types referenced is not obvious from viewing the DATA BASE MENU. The value is provided by an analyst familiar with how the input is used by the system. The chart in Figure 4-6 is used to consider these factors and designate the processing function category [ZWAN84]. According to the chart, three data element types (DET) and one referenced file type (FTR) implies a low level of information processing function. An analyst may intuitively adjust the result up or down one level by considering other factors. A detailed description of the procedure to classify function point parameters is given in each of the three references cited in Appendix E, Additional Sources for Function Point Training.

	1-4 DET	5-15 DET	16+ DET
0-1 FTR	L	L	A
2 FTR	L	A	H
3+ FTR	A	H	H

where

DET = data element types

FTR = file types referenced

L = low

A = average

H = high

Figure 4-6. Chart Used to Determine the Level of Information Processing Function For External Input Types.

Once parameters are identified, classified, and counted, the values are entered to the BYL Function Point Subsystem Screen, similar to the one shown in Section 2.3.1, Figure 2-8. The BYL Summary Report, Table 4-7, shows categorization of function point parameters for the CATSS Sensitivity Model. Generally, most of the parameter types were designated as simple in terms of their complexity.

4.5.2 Degree of Influence of Processing Characteristics

The Summary Report in Table 4-7 also illustrates the ratings assigned to each processing characteristic using available guidelines [ZWAN84, ALBR84, GORD87]. The ratings which were entered to the BYL FP Screen (Figure 2-8) decreased the function point count by 15%. Half of the characteristics were rated as not present or having no influence in the CATSS Sensitivity Model.

4.5.3 Results and Conclusions

The BYL adjusted function point total for CATSS was 213.35. Multiplying the function point total by the default compiler coefficient for FORTRAN (105) yielded a size estimate of 22,402 SLOC. The estimate is more than twice the actual system size of 9,177 (144% relative error).

The disparity in the actual versus estimated sizes has not been attibuted to any single factor. However, the fact that the estimated size was much larger does not lend credence to the premise that function points do not adequately account for code associated with scientific formulation (reference Section 2.3). If this were the case, an estimate that was much smaller than the actual size would have been obtained.

TABLE 4-7

BYL FUNCTION POINT REPORT FOR THE CATSS SENSITIVITY MODEL

********** SOFTWARE COST MODEL **********

Description:

FUNCTION COUNT & COMPLEXITY **

	Simple	Average	Complex	Unadjusted Function Points
External Input/Inquiry	33x3	6x4	2x6	135
External Output	13x4	1x5	0x7	57
Logical Internal File	6x7	1x10	0x15	52
External Interface File	0x5	1x7	0x10	7
Total unadjusted function points :				251

PROCESSING COMPLEXITIES **

Characteristic	Rating	Adjustment Factor
Data Communications	Strong	+2.5%
Distributed Functions	None	-2.5%
Performance	None	-2.5%
Heavily Used Config	None	-2.5%
Transaction Rate	None	-2.5%
Online Data Entry	Strong	+2.5%
End User Efficiency	Average	+0.5%
Online Update	None	-2.5%
Complex Processing	Average	+0.5%
Reuseability	Insignificant	-1.5%
Installation Ease	Moderate	-0.5%
Operational Ease	None	-2.5%
Multiple Sites	None	-2.5%
Facilitate Change	Insignificant	-1.5%

Net adjustment factor . +-15.0%
Total adjusted function points : 213.35

Compiler : FORTRAN Coefficient : 105
Estimated delivered source instructions : 22402

4.6 Application of the SPQR SIZER/FP Approach

To apply SPR's approach for estimating function points, a demonstrator disk of SPQR/20 was obtained. SPQR/20 (Version 1.1) is a general purpose estimating tool that predicts the cost and effort to develop software. SPR's implementation of function points was introduced within the SPQR/20 model in January 1986 and has since been reintroduced as a separate tool in SPQR SIZER/FP. Function point inputs in both models are the same as are the underlying equations and resulting estimates. As such, conclusions that are based upon the general approach are valid for both SPQR models referenced in this report.

There are two primary differences between the BYL implementation of function points described in the previous section and the SPR approach:

1. Function types are not classified as simple, average or complex.

2. A different approach is used to adjust the initial FP total for processing complexity.

The following procedure describes the general approach as it was applied to size the CATSS Sensitivity Model.

4.6.1 Procedure

SPQR requires eight primary inputs in order to generate an estimate. Appendix D describes the process to identify FP parameters for the CATSS Sensitivity Model. Counts for each of the five parameter types are used to obtain an unadjusted total. Three additional parameters define the complexity adjustment factor. On a scale of 1 to 5, logical complexity of CATSS was assigned a four (some difficult or complex calculations) and code complexity and data complexity were each assigned a value of three (well structured code with multiple files, switches, and data interactions).

4.6.2 Results and Conclusions

Figure 4-7 summarizes how inputs are used to obtain a function point estimate. The function point total is obtained with less effort than the Albrecht methodology that is implemented by the BYL model. Parameter counts are weighted accordingly and summed to obtain the unadjusted total.

Complexity adjustment increased the unadjusted total by 10%. The SPQR adjusted total for CATSS was 342 which, when contrasted to the BYL total of 213.35, is a relative difference of about 60%. This contradicts assessment that the total function point value produced by SPQR is within 2% of the current Albrecht methodology. Using a FORTRAN default language expansion factor of 105, the resulting size estimate produced was 35,910 SLOC. Based on CATSS actual size of 9,177 SLOC, the estimated size is nearly four times greater (291% relative error).

FUNCTION POINTS INPUTS

INITIAL ESTIMATE

ALBRECHT FUNCTION POINTS

NUMBER OF INPUTS	14	X	4	=	56
NUMBER OF OUTPUTS	14	X	5	=	70
NUMBER OF INQUIRIES	27	X	4	=	108
NUMBER OF DATA FILES	7	X	10	=	70
NUMBER OF INTERFACES	1	X	7	=	7
	UNADJUSTED TOTAL				311
	COMPLEXITY ADJUSTMENT				1.10
	ADJUSTED TOTAL				342

MOVEMENT KEYS: HOME, PAGE UP, PAGE DOWN, ↑

Figure 4-7. SPQR Function Point Estimate for the CATSS Sensitivity Model.

4.7 APPLICATION OF THE ASSET-R MODEL

IITRI was selected as a beta-test site for the ASSET-R model which was how it was obtained for the case study to size the CATSS Sensitivity Model. The most difficult aspect of applying ASSET-R, and one which sets it apart from others in this evaluation, is determining the value of one of its inputs. The analyst must estimate the number of operators and operands for the system that is to be sized.

4.7.1 Counting Unique Operators and Operands

The operator/operand count is a measure of the amount of code that is used for mathematical formulation. Operators are delimiter symbols or keywords in the engineering equations developed for the system. Examples include character string concatenation (//), exponentiation (**), division (/), addition (+), grouping (()), replacement (=), comparison (.GE.), and logical conjunction (.AND.). Also counted, in FORTRAN Statement Syntax, are DATA statements, EQUIVALENCE statements, and logical IF statements [SAP]. Operands are constants or variables contained within the engineering equations upon which an operation is being performed. Hence, in the simple equation: $A + B = C$; A, B, and C are operands while + and = are the operators.

Ideally, operator and operand counts are obtained from identified engineering formulation to be used within the system and any accompanying simulation software. In the case of the CATSS effort, simulation software was never a forerunner of actual implementation. Also, equations to be used in the CATSS Sensitivity Model were described only at a very high level in the project related documentation.

For ASSET-R application, the operator/operand count was obtained for CATSS using two different approaches:

1. Use an automated tool to count unique number of operators and operands from available CATSS source code.

2. Manually count the number of equations cited in project related documentation.

In the first approach, project personnel used an existing automated tool developed by NASA that accumulates Halstead measures (See Section 2.4) from

FORTRAN source code. NASA's Source Analyzer Program (SAP) accumulates metric data as each statement is analyzed. Measured quantities include unique operators, unique operands, and total operands.

An Air Force system called the Automated Measurement Systems (AMS) utilizes the SAP program. It was through application of the AMS on CATSS source code that the number of unique operators and operands was obtained. For the CATSS Sensitivity Model the number of unique operators = 3,651; unique operands = 5,157. Adding the two counts yields a total of 8,808 unique operators and operands.

The total count obtained using the AMS is a code-based value, whereas the count required for ASSET-R is specification-based. RCI data suggests the difference between specification and code-based counts is in the area of 1 to 4/4.5 for FORTRAN. That is, an equation listed in a specification document takes about four times the number of operators and operands to implement in FORTRAN source code. Hence, in defining a range of values for input to the ASSET-R model, Reifer suggests that the code-based total (8,808) be used for the maximum count. The average and minimum counts are derived by dividing the code-base total by the operator/operand expansion rate for the implementation language. The average count of operators and operands for CATSS is 2,202 (8,808/4). The minimum value is 1,957 (8,808/4.5).

The second approach used to obtain the operator/operand count involved counting the number of algorithms used by the system. ASSET-R provides a worksheet screen, shown in Figure 4-8, for the purpose of assisting the user in ascertaining unique operator/operand count. The worksheet defines five classes of operators and operands. The user has the option of entering the number of (1) basic algorithms, in lieu of (2) operands, (3) basic operators, (4) keyword operators, and (5) special operators.

The Technical Report for the CATSS Sensitivity Model [CATS86d] was obtained and all cited equations were counted. A total of 29 algorithms counted was used as input to the operator/operand worksheet (Figure 4-8).

Two size estimates were obtained in the procedure using the two values obtained for operator/operand count as input. All other inputs to ASSET-R were the same for both applications of the model.

DEMO - CO
OPERANDS/OPERATORS WORKSHEET

	MAX	AVG	MIN
1) Algorithms:	55	55	55
2) Operands:	129	129	129
3) Basic operators:	8	8	8
4) Keyword operators:	0	0	0
5) Special operators:	0	0	0
TOTALS:	192	192	192

Enter the number of functional algorithms used by the system to mechanize the computational formulation. Do not continue unless algorithms consume most of the processing internal to the system.
F2 - COUNTING CONVENTIONS ——— F8 - SIZING FACTORS

Figure 4-8. ASSET-R Operators/Operands Worksheet Screen.

4.7.2 Procedure

A possibility of nine different factors are used to calculate function points for ASSET-R. Each input is entered as a range of estimated values which are used to compute a weighted average.

Five of the nine factors - external inputs, outputs, inquiries, interfaces, and internal files - correlate to parameter inputs defined in the standard Albrecht methodology. Appendix D discusses how each of these factors was identified and counted. Since analysts were fairly certain of the totals for each parameter, the minimum, average, and maximum counts were each assigned the same value.

Two of the size factors, rendezvous and stimulus/response relationships, were determined not to apply to the CATSS effort and were set to zero.

Of the two remaining size inputs, the number of unique operators and operands were determined through application of an automated measurement tool and from project related documentation as discussed in the preceeding section. Two unique modes of operation were counted for the last factor. There is one interactive dialogue mode between the user and the CATSS system

as well as one execution processing mode. The project estimate summary report in Table 4-8 summarizes factor inputs for the CATSS effort.

Twelve of the remaining attributes are used to calibrate the function point estimate. Listed under ASSET-R Calibration parameters in Table 4-8, each of these attributes were assigned a rating from low to very high by the CATSS program manager. Half of the calibration factors were set to nominal indicating a standard environment.

The remaining three inputs described specific attributes about the effort. The CATSS Sensitivity Model is a scientific system with centralized system architecture using a single processor. The implementation language is FORTRAN.

All of the input parameters were entered to the ASSET-R model via input screens. Help windows and worksheet screens were used to quantify the estimates.

4.7.3 Results and Conclusions

The summary report in Table 4-8 shows the estimated size projected for the CATSS Sensitivity Model using the operator/operand count obtained through application of the AMS as input. The estimated size of 11,943 SLOC represents a relative error of 30% based upon the actual size of 9,177 SLOC.

In the second application of the model, a total of 29 algorithms counted was used as input to the operator/operand worksheet (Figure 4-8). With all other inputs set to the original CATSS input values, the size estimate obtained was 6,622 SLOC. Relative error, using algorithms as input, was 28%.

Effective use of the ASSET-R to size scientific and real-time systems will be contingent on the analyst's ability to obtain definitive data on the system's engineering formulation. **If credible results are to be obtained through this method, detailed engineering formulation must be provided by systems engineers as part of the system specification.**

Generally, the ASSET-R package was found to be extremely user-friendly. It is probably one of the more sophisticated sizing models in terms of its mathematical formulation. The Users Manual is well-written and easy to follow.

TABLE 4-8

ASSET-R SUMMARY REPORT FOR THE CATSS SENSITIVITY MODEL

ASSET-R Project Information

Project name:	CATSS
Estimate number:	0
Estimate date:	07/17/87

ASSET-R Base Estimate:

Source lines of code (SLOC's):	11943.20
Programming language:	FORTRAN
Mean number of functions:	75.40
Operand/operator count:	6473.79

ASSET-R Calibration Parameters:

Product complexity:	High
Requirements volatility:	Nominal
Degree of optimization:	Low
Degree of real-time:	Nominal
Degree of reuse:	Low
Database size:	Nominal
Use of modern programming techniques:	Nominal
Use of software tools/environments:	Nominal
Analyst capability:	High
Applications experience:	Very High
Environment experience:	Nominal
Programming language experience:	High

ASSET-R Size Factors:

	MAX	AVG	MIN
External Inputs:	14	14	14
External Outputs:	14	14	14
Internal Files:	7	7	7
Operating Modes:	2	2	2
Operands/Operators:	8808	2202	1957
Rendezvous:	0	0	0
Stimulus/Response:	0	0	0
External Inquiries:	27	27	27
External Interfaces:	1	1	1
Totals:	8873	2267	2022

ASSET-R Other Factors:

Type of software system:	Scientific System
Software system architecture:	Centralized
Architectural factor:	1.00
Language expansion factor:	105.00
Technology factor:	0.83

4.8 APPLICATION OF PRICE SZ

Access to PRICE SZ was attained through the Air Force Cost Center. Analysts familiar with the CATSS Sensitivity Model derived the inputs to be used in the model's application. The input parameters were sent to Wright Patterson AFB where Air Force personnel ran the Sizer and returned the resultant sizing estimate.

4.8.1 Procedure

Model inputs are summarized on the PRICE SZ output report shown in Table 4-9. Inputs are categorically defined as qualitative descriptors, quantitative descriptors, and sizing factors.

Qualitative descriptors characterize the development environment of the application. These were relatively easy to define in view of the fact that the CATSS effort was developed in-house and the engineers who designed and developed the software were accessible. Three of the parameters: system design skill, program design skill, and coding skill which refer to the skill and experience level of the software team will not be required input for the next release of PRICE SZ.

The sizing factors used were values that exhibited the "average" development environment (size calibration = 1) for software implemented in FORTRAN (language expansion = 5.5). Target size is a mandatory input for calibrating the model to the user's organization. It is set to zero for non-calibration. By not calibrating the model, PRICE engineers maintain that the model will produce estimates within 30% of the actual size, 50% of the time. With successive use, size that is within 10% (aggregate) of the actual is attained with calibration [RAPP85].

The qualitative inputs from the onset, were difficult for IITRI personnel to derive. After several exchanges with PRICE engineers, ambiguities seemed to be clarified and the inputs were sent to Wright Patterson AFB. The resulting PRICE SZ estimate, 64,618 SLOC for the CATSS Sensitivity Model which is actually 9,177 SLOC, prompted a review of the application.

TABLE 4-9

PRICE SZ OUTPUT SUMMARY REPORT

--- PRICE SOFTWARE SIZING MODEL ---

DATE 30-JUL-87 TIME 09:34 FILENAME : TERMINAL

CATSS SENSITIVTY MODEL

QUALITATIVE DESCRIPTORS

SYSTEM DESIGN SKILL	High	PROGRAM DESIGN SKILL	Average
CODING SKILL	Low	PROGRAM APPLICATION	Military
SYSTEM INTEGRATION	No	DESIGN REVIEWS	Yes
CODE WALK THRU	No	TOP DOWN APPROACH	Yes
STRUCTURE/MODULE TEST	Yes	REQUIREMENTS GROWTH	5%

QUANTITATIVE DESCRIPTORS

OUTPUT PAGES	0.0	ALPHANUMERIC DISPLAYS	11.0
GRAPHIC DISPLAYS	10.0	INPUT STREAMS	7.0
OUTPUT STREAMS	4.0	SYSTEM STATES	2.0
MESSAGE FIELDS	0.0	OPERATOR ACTIONS	92.0
INPUT ANALOGS	0.0	COMPUTED TABLES	1.0
FUNCTIONAL BULKINESS	1.0		

SIZING FACTORS

SIZE CALIBRATION	1.00	LANGUAGE EXPANSION	5.50
TARGET SIZE	0		

SIZING ESTIMATE

SOURCE INSTRUCTIONS:

LOW	19007
CENTER	21410
HIGH	24208

MACHINE INSTRUCTIONS:

CENTER	117752

After project personnel analyzed all of the initial input parameters it was decided that several of the PRICE SZ inputs derived were incorrect and those would have had considerable impact on the size estimate. Six of the quantitative descriptors were modified:

1. Number of graphic displays (from 14 to 10)
2. Number of input streams (from 15 to 7)
3. Number of output streams (from 51 to 14)
4. Number of system states (from 3 to 2)
5. Number of message fields (from 1 to 0)
6. Number of computed tables (from 5 to 1).

Inputs were also discussed at length with an RCA PRICE representative, Mr. Jim Otte, who further clarified parameter definitions which led to two additional changes:

1. Number of alphanumeric displays (from 25 to 11)
2. Number of output streams (from 14 to 4).

For the most part, the initial inputs reflected a too detailed analysis of the system. For example, each set of areal feature data used for CCM analysis is composed of seven separate files. Six of these files contain actual feature information: slope, soil type, stem diameter, stem spacing, surface roughness, and vegetation roughness. Originally, project personnel counted each feature as a separate input stream. In the second application, the six input streams were counted as one.

Similarly, project personnel were aware of five unique software created tables used for various calculations. It was shown that four of the five tables would not be evident from a user's perspective; only an engineer thoroughly familiar with the application would realize their need. Hence, the number of computed tables was reduced from five to one.

Parameter descriptions in the PRICE SZ Reference Manual [RCA85] in two cases were contrary to instruction received from PRICE engineers. The first case refers to the counting of alphanumeric displays. The Reference Manual states that the model considers one alphanumeric display to consist of 24 lines per display. Thus, a screen display servicing a single function was often counted as two or more alphanumeric displays depending upon its length - if length was greater than 24. Otte recommended counting alphanumeric displays on a <u>per function</u> basis. By disregarding one display consists of 24 lines, the number of alphanumeric displays was reduced from 25 to 11.

In the second case of misleading parameter descriptions, the Reference Manual states that the number of output streams should also include the number of unique processing tasks required to drive the output pages, alphanumeric displays, and the graphic displays. The instruction, in effect, misled project personnel into counting these processing tasks twice: once in the counts for output pages, alphanumeric, and graphic displays and secondly, in the total count for the number of output streams.

4.8.2 Results and Conclusions

The summary report in Table 4-9 shows the estimated size projected for the model to be 21,410 SLOC. The estimated size represents a relative error of 133% based upon the actual size of 9,177 SLOC.

The application of PRICE SZ to the CATSS Sensitivity Model emphasizes the fact that training to gain a better understanding of model inputs is required for Sizer's use. The PRICE SZ Reference Manual is currently being rewritten to clarify input parameter definitions. Also, through a learning process, analysts will likely obtain better estimates as more applications are sized.

5.0 CONCLUSION

The purpose of this study was to identify and evaluate currently available software sizing models that could be used to support the Air Force software acquisition process. The report initially focused on a number of automated and non-automated sizing techniques which were grouped into general categories in order to provide an overview of general approaches to sizing. The approaches were categorically defined as follows:

- Sizing by analogy
- Size-in-size-out
- Function point analysis
- Comparison of project attributes
- Linguistic approach
- Other

A more detailed evaluation focused on nine of the fourteen identified models that are fully automated. To gain additional insight into the strengths and weaknesses of each technique, six of the nine models were applied to size the CATSS Sensitivity Model. The results produced through application of the selected techniques to the same system are shown in Table 5-1. The actual size of the CATSS Sensitivity Model in 9,177 SLOC.

TABLE 5-1

SUMMARY OF MODEL SIZE ESTIMATES FOR THE CATSS SENSITIVITY MODEL

MODEL	SLOC
ESD	37,600+
SPQR	35,910
BYL	22,402
PRICE SZ	21,410
ASSET-R	11,943
SSM	11,700
ASSET-R	6,622

Note: The ESD estimate does not include sizes for three of the twelve CATSS modules. Also, two estimates for ASSET-R reflect two different approaches used to derive model input.

Throughout the evaluation process, various capabilities, dependencies, and considerations were noted for different models. A set of usage criteria for describing each model provided a common framework for noting the advantages and deficiencies of using one technique versus another. By this criteria, it was concluded that

- The analogy approaches (ESD, SSA, and QSM Fuzzy Logic) are the easiest models to apply in terms of the few inputs and minimal training they require.

- QSM Size Planner is the only model with a built-in historical data collection and maintenance capability that supports sensitivity analysis. CEIS supplies an interface to the Mainstay Data and Analysis System which supports collection and sensitivity analysis but it is a separate product.

- The function point approaches (ASSET-R, BYL, QSM Function Points, and SPQR), PRICE SZ, and QSM Fuzzy logic require the least knowledge or experience in the application area to be effective.

- ASSET-R, CEIS, PRICE SZ, QSM Fuzzy Logic, QSM Standard Component Sizing, and SSM are applicable in virtually all user environments. ESD and SSA are limited to applications comparable to those described in their respective size databases. The BYL, QSM, and SPQR function point implementations have not been validated for scientific or real-time systems (See Section 2.3). The function point approach should be further evaluated for these types of systems and/or studies which address performance of these models in scientific or real-time environments should be made available.

- ASSET-R and QSM Size Planner are the most sophisticated models in terms of their output.

- BYL and ASSET-R are ranked highest in terms of user-friendly interface and user support.

- The analogy approaches (SSA, ESD, and QSM Fuzzy Logic), CEIS, and SSM can be applied earliest in the life-cycle during software requirements analysis.

- ASSET-R and SSM provided the most accurate size estimates in the test case study which is described in Section 4.

APPENDIX A

MODEL INPUT PARAMETERS: DETAILED DEFINITIONS

This appendix contains a detailed description of model input parameters for automated models listed in Section 3.0. The input parameter descriptions are listed by model name.

(1) ASSET-R
(2) BYL
(3) CEIS
(4) ESD Software Sizing Package
(5) PRICE SZ
(6) QSM Size Planner
(7) SPQR SIZER/FP
(8) SSA
(9) SSM

Input variables for each model are subdivided by input type as follows:

- *Identification inputs* - data used for identification only and not used for estimating.
- Quantitative inputs - inputs that require determination of quantity.
- Qualitative inputs - inputs that are used to characterize the development environment and complexity of the application.
- Modular (functional) components - inputs that require knowledge of the modularity and functionality of the software system.
- Calibration factors - correction factors that reflect a development environment which would cause deviation from an estimate for a "typical" development environment.

The description for each parameter contains the following information:

- Name of the input variable
- Type of input - Input value formats are categorized as follows:

R = Rating
% = Percentage or proportion
# = Count
C = Category
S = Character string
Y/N = Yes or No
E = Empirically derived

- Default value

- Required or optional input

- Input variable definition - Descriptions provided here are summary explanations. More detailed definitions and additional clarification of inputs are provided in the training for use of the model or in the model's reference/user manual.

The format for each variable description is as follows:

Name [type, default, required or optional] - input variable definition.

ASSET-R

Identification Inputs

1. **Project name** [S, No default, Required] - project identifier and file name to store the project data on disk.

2. **Date** [S, Default = current date, Required] - date of performance of the estimate.

Quantitative Inputs

Note: Each of the nine quantitative sizing factors are entered to the ASSET-R model as a range of estimates: maximum, average, and minimum.

1. **External inputs** [#, No default, Required] - unique data and/or control inputs that cross the external boundary of the system and cause processing to happen. Input types include input files or tables, input forms, input screens, and input transactions.

2. **External outputs** [#, No default, Required] - unique data or control outputs that leave the external boundary of the system after processing has occurred. Types include output files or tables, output screens, output reports, and output transactions.

3. **Internal files** [#, No default, Required] - logical groupings of data and/or control information which are to be stored internal to the system. Types include databases, logical files, control files, and directories.

4. **Operating modes** [#, No default, Required] - unique modes of operation which are performed internal to the system. Types are dialogue (batch, interactive, and interpretive), processing (diagnostic, calibration, and execution), and error modes (checkpoint/restart and reconfiguration).

5. **Operands/operators** [#, No default, Required] - a measure of the size consumed by mathematical equations specified for the system. The value is specified by entering either the number of

   operands (unique constants and variable names),

   basic operators (+, -, *, /, exponentiation, etc.),

   keyword operators (if-the-else, do-while, begin--end, etc.), and

   special operators (macro functions, entry points, trace functions, etc.)

   or number of **algorithms** if preceeding values cannot be determined.

Note: S = Character string, # = Count.

6. **Rendezvous** [#, No default, Required] - a measure of concurrency in the system. Types are concurrent processes, concurrent tasks, and concurrent processors.

7. **Stimulus/response relationships** [#, No default, Required] - different timelines handled by the operating modes that are part of the system. Types are causal relationships which cause switchover from one function to another and temporal relationships which cause switchover from one mode to another.

8. **External inquiries** [#, No default, Required] - input/output queries which require an immediate output. Types include prompts, interrupts, and calls.

9. **External interfaces** [#, No default, Required] - unique files/programs passed across the external boundary of the system. Types include common utilities, math libraries, program libraries, shared databases, and shared files.

<u>Qualitative Inputs</u>

1. **Type of software system** [C, Default = other, Required] - application type which best describes the system. Types are avionics, command & control, data processing, scientific, simulation, support software, telecommunication, test, or other.

2. **Software system architecture** [C, Default = centralized with single processor, Required] - architecture of the software system. Types are centralized (single processor), tightly-coupled (multiple processor), loosely-coupled (multiple processor), federated (multiple processor communicating via system bus or communications channel), distributed (centralized database), distributed (distribued database), or other.

3. **Requirements volatility** [R, Default = nominal, Required] - rating from 1 (low - essentially no changes) to 5 (extra high - < 30% nonchanging requirements) for volatility of system requirements.

4. **Programming language** [R, Default = FORTRAN (100%), Required] - programming language(s) used to develop the software and percentage of each.

5. **Product complexity** [R, Default = nominal, Required] - rating from 1 (very low - simple code) to 6 (extra high - time dependent/stochastic math) for product complexity.

6. **Degree of optimization** [R, Default = nominal, Required] - rating from 1 (low - optimization level < 50%) to 5 (extra high - optimization level = 100%) indicating whether software is memory, speed, and input/output limited.

Note: # = Count, C = Category, R = Rating.

7. **Degree of real-time** [R, Default = nominal, Required] - rating from 1 (low - batch response) to 5 (extra high - multi-tasking with nanosecond response range) indicating system performance requirements.

8. **Degree of reuse** [R, Default = nominal, Required] - rating from 1 (low - essentially no reuse) to 4 (very high - > 30% reuse) identifying software reuse level.

9. **Database size** [R, Default = nominal, Required] - rating from 1 (low - database size < 10% of program size) to 4 (very high - database size > 1000% of program size) identifying relative database size.

10. **Use of modern programming techniques** [R, Default = nominal, Required] - rating from 1 (very low - no use) to 5 (very high - heavy use) indicating sophistication of programming techniques used.

11. **Use of software tools/environments** [R, Default = nominal, Required] - rating from 1 (very low - basic language tools) to 6 (extra high - extensive integrated environment) identifying the level for software tools and tool environments used in the development process.

12. **Analyst capability** [R, Default = nominal, Required] - rating from 1 (very low - bottom 15%) to 5 (very high - superstar, top 10%) indicating skill level of analysts involved in the development process.

13. **Applications experience** [R, Default = nominal, Required] - rating from 1 (very low < 4 months experience) to 5 (very high - > 6 years experience) indicating the amount of experience the analysts have with the types of applications within the software system.

14. **Environment experience** [R, Default = nominal, Required] - rating from 1 (very low - < 1 month experience) to 4 (high - > 1 year experience) indicating the amount of experience the analysts have with the software development environment.

15. **Language experience** [R, Default = nominal, Required] - rating from 1 (very low - < 2 months experience) to 5 (very high - > 2 years experience) indicating the amount of experience the analysts have with the programming language(s).

Calibration Factors

Note: The ASSET-R package can be calibrated to different environments by editing its parameter file. The parameter file contains (2) adjustment factors which may be edited by the user:

1. **Language expansion factor** [E, Default, Not required] - TABLE A-1 provides current values that correlate language source code to function points. Correlation coefficients show the fit to actuals in the RCI database.

2. **Architectural constant** [E, Default, Not required] - TABLE A-2 provides the table of values for the seven architectural types.

Note: R = Rating, E = Empirically derived.

TABLE A-1

ASSET-R LANGUAGE EXPANSION FACTORS

LANGUAGE	SLOC's PER FUNCTION POINT	CORRELATION COEFFICIENT
Ada	72	0.887
APL	38	0.896
ASSEMBLER	400	0.751
C	90	0.776
CHILL	106	0.734
CMS-2	80	0.871
COBOL	100	0.913
FORTRAN	105	0.899
JOVIAL	80	0.698
Pascal	70	0.743
PL/1	65	0.911
PROLOG	64	0.753
Other high level language	86	N/A

TABLE A-2

ASSET-R ARCHITECTURAL CONSTANTS

ARCHITECTURE	VALUE
• Centralized	1.0
• Tightly-coupled, multi-processor	1.3
• Loosely-coupled, multi-processor	1.5
• Federated	1.6
• Distributed with centralized database	1.8
• Distributed with a distributed database	2.0
• Other	1.0

Identification Inputs

1. **Data set name** [S, No default, Required] - user assigned file name that contains the set of data used to derive a project estimate.

2. **Description** [S, No default, Required] - description of the selected data set.

Quantitative Inputs

1. **External input/inquiry types** [#, No default, Required] - unique number of user inputs of data or controls and user inquiries which require a response to be generated. An external input/inquiry type should be counted if it has a different format than other external input/inquiry types or if it is anticipated to necessitate distinct processing.

   Each external input/inquiry type should be classified within three levels of complexity, as follows:

   - Simple — Data types are minimal and few logical internal files need be accessed. The user interface has little influence over the selected external input/inquiry type.

   - Average — The external input/inquiry falls between the definitions given for the simple and complex.

   - Complex — The external input/inquiry is a composite of a variety of data types. The user interface plays a major role in the design of the external input/inquiry type. Responses to inquiries usually require major conversions with many files (internal or external) being accessed.

2. **External output types** [#, No default, Required] - unique number of distinct data or signal output types that are extended beyond the online processing capability of the application being measured. An external output type should be counted if it has a different format than other external output types or if it is anticipated to necessitate distinct processing from other external data types of similar or identical format.

   Each external output type should be classified within three levels of complexity, as follows:

   - Simple — Data types are minimal and few logical internal files need be accessed. The user interface has little influence over the selected external output type. Reports require little data conversion and usually have only one or two columns.

Note: S = Character String, # = Count

- Average — The external output type falls between the definitions given for the simple and complex. Reports require the conversion of several data types and tend to involve several columns with totals or other column processing.

- Complex — The external output type is a composite of a variety of data types. The user interface plays a major role in the design of the external output type. Reports require major data conversions or cross-correlation of many files (internal or external).

3. **Logical internal file types** [#, No default, Required] - unique number of distinct logical groups of data or control information. A logical internal file type should be counted if it has a different format than other logical internal file types or if it is separately generated, used, or maintained by the softwared product. Logical files should not be confused with physical files.

   Each logical internal file type should be classified within three levels of complexity, as follows:

   - Simple — There are minimal record and data element types. Considerations for recovery and performance considerations are also minimal.

   - Average — The logical internal file type falls between the definitions given for the simple and complex.

   - Complex — There are many record and data element types. There are also major performance and recovery considerations.

4. **External interface file types** [#, No default, Required] - unique number of files or other groups of user data or control information that is passed or shared between multiple applications. An external interface file type should be counted if it has a different format than other external interface file types or if it is separately received or sent.

   Each external interface file type should be classified within three levels of complexity, as follows:

   - Simple — There are minimal record and data element types. Considerations for recovery and performance considerations are also minimal.

   - Average — The external interface file type falls between the definitions given for the simple and complex.

   - Complex — There are many record and data element types. There are also major performance and recovery considerations.

Note: # = Count

5. **New delivered source instructions** [E, No default, Required] - new lines of code.

6. **Adapted delivered source instructions** [E, No default, Required] - lines of code brought over from software written for the same underlying virtual machine.

7. **Converted delivered source instructions** [E, No default, Required] - lines of code rendered from software from a different underlying virtual machine.

<u>Qualitative Inputs</u>

Note: Fourteen processing characteristics are rated according to the degree of influence that each has on the software application. The degree of influence ratings which are applied to the processing characteristics are:

- None — The characteristic does not apply, is not present, or has no influence.
- Insignif — The characteristic has an insignificant influence on the value of the application to the targeted end user(s).
- Mod — The characteristic has a moderate influence on the value of the application to the end user(s).
- Avg — The characteristic has an average influence on the value of the application to the end user(s).
- Signif — The characteristic has a significant influence on the value of the application to the end user(s).
- Strong — The characteristic has a strong, permeating influence on the value of the application to the end user(s).

The 14 processing characteristics and their meanings follow:

1. **Data communications** [R, No default, Required] - refers to data and control information that are a part of the application and are being transmitted or received over communication facilities, including terminals.

2. **Distributed functions** [R, No default, Required] - refers to distributed data or processing functions.

3. **Performance** [R, No default, Required] - refers to performance objectives in terms of the amount of influence that throughput and/or responsiveness have on the design, development, installation, and support of the application.

Note: E = Empirically derived, R = Rating.

4. **Heavily used configuration** [R, No default, Required] - refers to the complexity of hardware and communication lines and to the extent to which these resources are previously committed.

5. **Transaction rate** [R, No default, Required] - refers to the presence of a high transaction rate and to the extent to which that rate affects the design, development, and installation of the application.

6. **Online data entry** [R, No default, Required] - refers to the handling of online data functions.

7. **End user efficiency** [R, No default, Required] - refers to the applications emphasis on end user efficiency.

8. **Online update** [R, No default, Required] - concerns any online capability to update the logical internal files.

9. **Complex processing** [R, No default, Required] - refers to the necessary presence of complex processing and the influence it has on the design, development, and installation of the application. Complex processing includes reentrant code, recursion, interrupt handling, complex algorithms, detailed I/O handling, dynamic space allocation, etc.

10. **Reuseability** [R, No default, Required] - refers to the reuseability of the code in the application in other applications or other host environments.

11. **Installation ease** [R, No default, Required] - refers to the amount that installation ease affects product design and the ability to verify installation procedures across the spectrum of likely host environments.

12. **Operational ease** - [R, No default, Required] - refers to the variety of features, including back-up and restore procedures, program fluidity, recovery procedures, and general ease of use.

13. **Multiple sites** [R, No default, Required] - the software product has been designed, developed, and supported for implementation in multiple sites or multiple types of users.

14. **Facilitate change** [R, No default, Required] - refers to the accessibility of file formats, data and processing details of the software product provided in the design, and development of the software to facilitate future change.

Note: R = Rating.

Calibration Factors

1. **Compiler** [S, No default, Required] - refers to any compiler, interpreter, code generator, preprocessor, assembler, etc. that acts upon the source instruction.

2. **Compiler coefficient** [E, No default, Required] - a measure of the power of the compiler. The value is derived based upon the following formula for a previously developed project:

   Delivered Source Instructions/Function Point Measure.

Note: S = Character string, E = Empirically derived.

Identification Inputs

1. **Attribute names** [S, Default = see below, Required] - set of six (5) attribute names and corresponding four-character abbreviations that are associated with task size. The current proposed default list is:

• Complexity	CPLX
• Peak Staff	STAF
• Technology Rating	TECH
• Requirements Volatility	RVOL
• Specifications Level	SPEC
• Required Reliability	RRLY.

2. **Reference task names** [S, No default, Required] - three (3) completed task names that are similar in nature to the task to be estimated.

3. **New task name** [S, No default, Required] - name of the task to be estimated.

Quantitative Inputs

1. **Reference task sizes** [E, No default, Required] - actual number of source lines of code in each reference task.

Qualitative Inputs

1. **Compare attributes** [R, Default = 1, Required] - pairwise compare the importance of one attribute over another. Once one attribute is determined to be more important than another, rate the importance according to the following list:

   1 = Equal Importance
   3 = Weak Importance
   5 = Strong Importance
   7 = Demonstrated Importance
   9 = Absolute Importance.

2. **Compare reference tasks** [R, Default = 1, Required] - pairwise compare the importance of one reference task to another for each of the attributes. Use the same importance rating list.

3. **Compare new task to reference tasks** [R, Default = 1, Required] - compare the importance of the new task to the reference tasks for each attribute. Use the same importance rating list.

Note: S = Character String, R = Rating, E = Empirically derived.

ESD SOFTWARE SIZING PACKAGE

Note: User input parameters are for retrieval of sizing data. The user may select entries from the ESD sizing database by one or a combination of two or more of the following criterion:

Modular (Functional) Components

1. **System name** [S, No default, Optional] - previously developed system of similar function and environmental requirements.

2. **Development computer** [S, No default, Optional] - the computer on which the software is to be developed.

3. **Language** [S, No default, Optional] - the language in which the software is to be developed. If multiple languages are to be used for development, language should be designated at a functional level.

4. **Index number** [E, No default, optional] - values predefined to correlate with specific software functions (modules) where software functions are units of code ranging in sizes from 2 to 500,000 SLOC. Software functions, which are the lowest level at which data exists in the ESD sizing database, fall into two basic types: operational and support functions. Table A-3 lists software function categories and associated indexes.

5. **Function name** [S, No default, Optional] - refers to the function name as it appears in the ESD database as opposed to the higher level function associated with an index.

6. **Development status** [C, No default, Optional] - indicates whether the size parameter for the database entry is estimated or actual. Status includes completed, test, code, design, or RQTM.

7. **Range of SLOC** [E, No default, Optional] - used to filter out entries with extreme sizes that the user may wish to preclude from statistical analysis that yields a most likely value for the new function.

Note: S = Character string, E = Empirically derived, C = Category.

TABLE A-3

SOFTWARE FUNCTION CATEGORIES

Operational

Category	No.	Function
Displays	1.1	Avionics
	1.2	Command, Control, Communications
	1.3	Acoustics
	1.4	Combat Control System
	1.5	Other
Avionics	2.1	Mission Planning
	2.2	Navigation
	2.3	Aircraft Steering and Flight Control
	2.4	Sighting, Designation, and Location Determination
	2.5	Weapon Delivery
	2.6	Electronic Countermeasures
	2.7	Other
Command, Control, and Communications	3.1	Network Monitoring
	3.2	Network Control and Switching
	3.3	Sensor Control
	3.4	Signal Processing
	3.5	Message Processing
	3.6	Message Distribution
	3.7	Message Logging and Retrieval
	3.8	Data Reduction
	3.9	Other
Executive	4.1	Computer Resource Management
	4.2	Computer Operator Interface
	4.3	Other Terminal Operator Interface
	4.4	Special Device Interface
	4.5	Other Input or Output
	4.6	Error Handling/Reconfiguration/ Recovery
	4.7	Multicomputer Configuration Control
	4.8	Performance Monitoring and Data Collection
	4.9	Other
Data Base	5.1	On-Line Retrieval and Output
	5.2	On-Line Initialization and Updating
	5.3	Other
Training	6.1	Control of Exercise Sequencing
	6.2	Operator Performance Data Collection
	6.3	Other
On-Line Equipment Diagnostic	7.1	System Readiness Test
	7.2	Computer Diagnostic
	7.3	Memory Diagnostic
	7.4	Display Diagnostic
	7.5	Switch/Indicator Panel Diagnostic
	7.6	I/O Diagnostic
	7.7	Mode Diagnostic
	7.8	Other
Combat Control System	18.1	System Management
	18.2	Target Motion Analyzer
	18.3	Weapons Interface
	18.4	Weapons Background Processing/Fire
Acoustics	19.1	Detection
	19.2	Tracking
	19.3	Classification
	19.4	Ranging
	19.5	Acoustic Support

Support

Category	No.	Function
Operating System	8.1	Computer Resource Management
	8.2	Computer Operator Interface
	8.3	Terminal Operator Interface
	8.4	Input or Output
	8.5	Error Handling/Reconfiguration/ Recovery
	8.6	Performance Monitoring and Data Collection
	8.7	Other
Equipment Maintenance	9.1	Off-Line Computer Diagnostics
	9.2	Other
Software Development	10.1	Higher Order Language Compiler
	10.2	Assembler
	10.3	Debugger
	10.4	Loader or Editor
	10.5	Other
Off-Line Data Base Management	11.1	Data Base Definition
	11.2	Data Base Initialization or Updating
	11.3	Retrieval and Output Formatting
	11.4	Data Base Restructuring
	11.5	Off-Line Data Base
	11.6	Other
Design	12.1	Data Base Design
	12.2	Data Base Processor Design
	12.3	Performance Simulation
	12.4	Data Reduction
	12.5	Data Analysis
	12.6	Other
Test Software	13.1	Test Case Generation
	13.2	Test Case Data Recording
	13.3	Test Data Reduction
	13.4	Test Analysis
	13.5	Other
Utilities	14.1	Media Conversions
	14.2	Format Translation
	14.3	Sort/Merge
	14.4	Program Library Maintenance
	14.5	Other
Off-Line Training	15.1	Data Reduction
	15.2	Training Analysis
	15.3	Scenario Preparation
	15.4	Other
Project Management	16.1	Project Event Status Accounting
	16.2	Schedule Maintenance/Projection
	16.3	Financial Accounting
	16.4	Software Cost Reporting
	16.5	Hardware Cost Reporting
	16.6	Software Cost Prediction
	16.7	Hardware Cost Prediction
	16.8	Other
Hardware Subsystem Simulations	17.1	Interfacing Hardware Simulations
	17.2	Environmental Simulations
	17.3	Operator Action Simulations
	17.4	Other

PRICE SZ

Identification inputs

1. **Project title** [S, No default, Required] - name of the project for which the estimate is being made.

Quantitative inputs

Note: Items 1 through 4 refer to output.

1. **Output pages** [#, No default, Required] - uniquely formatted pages of data output to a line printer; 1 page = 66 lines.

2. **Alphanumeric displays** [#, No default, Required] - unique alphanumeric displays; 1 page = 24 lines.

3. **Graphic displays** [#, No default, Required] - unique raster or other types of graphic display formats, X-Y plotting boards, and other real-time command and control devices that employ designs using pixels, aspect ratios, etc.

4. **Output streams** [#, No default, Required] - uniquely processed data files or streams of data that are output to peripheral devices or other interfaces.

Note: Items 5 through 8 refer to input.

5. **Input message fields** [#, No default, Required] - unique message fields, input data set variables, and interface messages required for operation of the software program.

6. **Operator actions** [#, No default, Required] - operator actions or discreet inputs caused from operator inputs

7. **Input analogs** [#, No default, Required] - continuous variables generally input by sensor devices such as radar, temperature, or other real-time input devices.

8. **Input streams** [#, No default, Required] - uniquely processed data files or streams of data that are received from peripheral devices or other interfaces.

9. **Computed or created tables** [#, No default, Required] - unique program created tables or variables used for various calculations.

10. **System states** [#, No default, Required] - number of modes of operation of the software program; i.e., training mode, test, and mission modes.

Note: S = Character String, # = Count.

Qualitative Inputs

1. **Program application** [C, No default, Required] - military or commercial application type.

Note: Items 2 through 4 refer to skill and experience of the software team. The next release of PRICE SZ will not include these input parameters.

2. **System design skill** [R, No default, Required] - rating from nominal to superior of the skill and experience level of the system engineering team developing design specifications from system requirements.

3. **Program design skill** [R, No default, Required] - rating from nominal to superior of the skill and experience level of the software design engineering team developing the detailed software design from design specifications.

4. **Coding skill** [R, No default, Required] - rating from nominal to superior of the skill and experience level of the programming team that will translate detailed software design into code.

5. **Integration with another system** [Y/N, Default = No, Required] - denotes whether software described is stand-alone.

6. **Design review** [Y/N, Default = No, Required] - denotes whether in-house customer design review(s) are to be accomplished.

7. **Code walk-thru** [Y/N, Default = No, Required] - denotes whether code walk-thrus are required.

8. **Top-down approach** [Y/N, Default = No, Required] - denotes whether top down design strategy is used where a large problem is decomposed into smaller, less complex problems.

9. **Structure/module approach** [Y/N, Default = No, Required] - denotes whether software is built and tested gradually as opposed to completely developed and then tested.

10. **Program requirements growth** [%, No default, Required] - rating from nominal (0%) to superior (18%) that describes amount of software growth anticipated as a result of changing or increased requirements.

Calibration Factors

1. **Functional bulkiness** [E, No default, Required] - describes software team experienced with language and availability of software tools. Normal value equals 1, more tools available result in values < 1, and new language or less tools available give > 1 values.

Note: C = Category, R = Rating, Y/N = Yes or No, % = Percentage,
E = Empirically derived.

2. **Size calibration factor** [E, No default, Required] - describes the user's organization specific know-how or the way an organization implements software projects. Normal value equals 1 with range expected from .7 to 1.5.

3. **Language expansion ratio** [E, No default, Required] - describes the combination of languages and compilers used in the development effort. For size output in machine level instructions (assembly), value = 1.

4. **Target size** [E, No default, Required for calibration mode] - total amount of MLI or HOL.

Note: E = Empirically derived.

QSM Size Planner

Identification Inputs

1. **Project name** [S, No default, Required] - the name of the project for which the estimate is being made.

Input variables for Size Planner are organized by sizing technique. Sizing functions for newly developed software are:

- Fuzzy Logic Sizing
- Function Points Sizing
- Standard Component Sizing.

Fuzzy Logic Sizing

Qualitative Inputs

1. **Application type** [C, No default, Required] - application type which best describes the system. Application types are microcode/firmware, real time, avionic, system software, command and control, telecom and message switching, scientific, process control, business, mixed application, or unknown.

2. **Size category** [C, No default, Required] - overall size range of the system to be developed. Size ranges are very small, small, medium, medium large, large, very large.

3. **Size range** [C, No default, Required] - the size range of the system within the broader **size category.** Size ranges are low, midlow, midhigh, and high.

Function Points Sizing

Quantitative Inputs

1. **Inputs** [#, No default, Required] - unique number of data files, control information, or input screens that enter a program from an external source. Inputs are counted if they require unique processing logic or introduce new formats.

2. **Outputs** [#, No default, Required] - unique number of data files, control information, or output screens that leave a program and go to an external source. Outputs are counted if they require unique processing logic or introduce new formats.

3. **Inquiries** [#, No default, Required] - unique input/output combination such as a HELP screen, selection menu, or inquiry where an input is entered to direct a search of internal files and generate an immediate output. Inquiries are counted if they require unique processing logic and cause no change to internal data.

Note: S = Character string, C = Category, # = Count.

4. **Internal files** [#, No default, Required] - number of input and output files which the program will generate, access, or update. Also, each hierarchical path through a database or table in a relational database that requires unique processing.

5. **Interfaces** [#, No default, Required] - files or databases between or shared among separate applications.

Each of the 5 function point parameters described above are rated according to 5 complexity levels. Complexity levels are:

- Very simple
- Simple
- Average
- Complex
- Highly complex.

Qualitative Inputs

1. **Primary language** [C, No default, Required] - primary language that the system is programmed in.

Standard Component Sizing

Quantitative Inputs

For each of the components that follow, the user must supply a range of estimates labeled:

- low,
- most likely, and
- high.

The twelve components are:

1. **Source statements** [#, No default, Optional]

2. **Object instructions** [#, No default, Optional]

3. **Bits** [#, No default, Optional]

4. **Bytes** [#, No default, Optional]

5. **Words** [#, No default, Optional]

6. **Files** [#, No default, Optional]

7. **Modules** [#, No default, Optional]

8. **Subsystems** [#, No default, Optional]

9. **Screens** [#, No default, Optional]

Note: # = Count, C = Category.

10. **Reports** [#, No default, Optional]

11. **Interactive programs** [#, No default, Optional]

12. **Batch programs** [#, No default, Optional]

Qualitative Inputs

1. **Primary language** [C, No default, Required] - primary language that the system is programmed in.

The following ratings should also be provided for each of the twelve standard components.

2. **Confidence level** [R, Default = 1, Required] - rating that indicates the level of confidence in the component as an indicator of the ultimate size of the system. Low confidence = 1, moderate confidence = 2, high confidence = 3.

3. **QSM DB weight** [%, Default is calculated, Required] - indicates the combination of historical data from the user and the QSM database to be used for estimating source statements from user estimates of each component.

New, Reused, Rebuilt Sizing

Note: The following inputs are entered when a considerable amount of existing software may be reused and/or modified for a newly developing system.

Quantitative Inputs

For each of the seven parameters that follow, the user must supply a range of estimates labeled:

- low
- most likely, and
- high.

The seven parameters are:

1. **New code** [E, Default based on combined weighted estimates from prior functions employed, Required] - the new lines of executable source code to be developed.

2. **Reused code** [E, No default, Required] - the lines of code in modules which will be reused, but modified by additions, changes, and deletions.

Note: # = Count, C = Category, R = Rating, % = Percentage,
E = Empirically derived.

3. **Added code** [E, No default, Required] - the lines of code which must be added to the reused modules.

4. **Changed code** [E, No default, Required] - the lines of code out of the modified but reused modules which must be changed.

5. **Deleted code** [E, No default, Required] - the lines of code which must be deleted from the reused modules.

6. **Removed code** [E, No default, Required] - the lines of code which correspond to modules or programs that are removed as whole entities.

7. **Tested code** [E, No default, Required] - the lines of code from the modified but reused modules which exist and require no modifications but do require testing with new and modified software.

Note: E = Empirically derived.

SPQR SIZER/FP

Identification Inputs

1. **Security level** [S, No default, Required] - user assigned security level such as COMPANY CONFIDENTIAL or NONE or TOP SECRET.

2. **Organization** [S, No default, Required] - the company, agency, or university for which the estimate is being made.

3. **Project name** [S, No default, Required] - the name of the project for which the estimate is being made.

4. **Manager** [S, No default, Required] - the name of the person responsible for the project being estimated.

5. **Current date** [S, No default, Required] - current date of performance.

6. **Comments** [S, No default, Required] - any remarks relevant to the sizing estimate.

Quantitative Inputs

1. **Inputs** [#, No default, Required] - unique number of input files, control information, or input screens that enter a program from an external source. Inputs are counted if they require unique processing logic or introduce new formats.

2. **Outputs** [#, No default, Required] - unique number of output files, control information, or output screens that leave a program and go to an external source. Outputs are counted if they require unique processing logic or introduce new formats.

3. **Inquiries** [#, No default, Required] - unique input/output combination such as a HELP screen, selection menu, or inquiry where an input is entered to direct a search of internal files and generate an immediate output. Inquiries are counted if they require unique processing logic and cause no change to internal data.

4. **Data files** [#, No default, Required] - number of input and output files which the program will generate, access, or update. Also, each hierarchical path through a database or table in a relational database that requires unique processing.

5. **Interfaces** [#, No default, Required] - files or databases passed between or shared among separate applications.

6. **Base code size** [#, No default, Required for enhancement/maintenance estimate type] - refers to existing code of a program or system. The number of source statements divided by 1000 (KLOC) of existing code excluding comments, JCL, and included code.

Note: S = Character string, # = Count.

Qualitative Inputs

1. **Estimate type** [C, No default, Required] - a new program, an enhancement, or a maintenance change estimate type.

2. **Base logical complexity** [R, No default, Required for enhancement/maintenance estimate type] - Rating from 1 (mostly simple algorithms) to 5 (many complex calculations) for logical complexity of the base system.

3. **Base code structure** [R, No default, Required for enhancement/maintenance estimate type] - rating from 1 (nonprocedural) to 5 (poor structure with many complex paths and modules) for base program structure.

4. **Base code data complexity** [R, No default, Required for enhancement/maintenance estimate type] - rating from 1 (simple data with few variables) to 5 (very complex data elements and data interaction) for complexity of both the file structure and the data elements which the program or system will utilize.

5. **New logical complexity** [R, No default, Required] - rating from 1 (mostly simple algorithms) to 5 (many complex calculations) for logical complexity of the new system.

6. **New code structure** [R, No default, Required] - rating from 1 (nonprocedural) to 5 (poor structure with many complex paths and modules) for new program structure.

7. **New code data complexity** [R, No default, Required] - rating from 1 (simple data with few variables) to 5 (very complex data elements and data interaction) for complexity of both the file structure and the data elements which the program or system will utilize.

Calibration Factors

1. **Base code language** [C, No default, Required for enhancement/maintenance estimate type] - refers to exisiting code of a program or system.

2. **Base code language level** [C or E, Default values are provided for each language, Required for enhancement/maintenance estimate type] - number of assembler language statements it will take to create the same function that one statement will take in the base code langugage.

3. **New code language** [C, No default, Required] - source language(s) of a new program or system.

4. **New code language level** [C or E, Default values are provided for each language, Required] - number of assembler language statements it will take to create the same function that one statement will take in the new code language.

Note: C = Category, R = Rating, E = Empirically derived.

Modular (functional) Components

1. **Platform** [C, No default, Required] - describes system deployment: G = ground, F = flight.

2. **Keyword** [E, No default, Required] refers to functional application of entries contained in the software sizing database. The total list of functions has been grouped into approximately 34 "standard" functions identified by keyword used for retrieval in a database search operation. Table A-4 lists keywords for SSA.

TABLE A-4

SSA STANDARD FUNCTIONS IDENTIFIED BY KEYWORD

ACQ	STATION ACQUISITION	MSGE	MESSAGE
APPL	APPLICATION SOFTWARE	MISC	MISCELLANEOUS FUNCTIONS
ATT	ATTITUDE DETERMINATION	HIMISC	MISCELL. FUNCTIONS (>15 K)
CKS	CHECKS	MONTOR	MONITOR, STATUS
CMD	COMMANDING, COMMAND GENERATION	ONLINE	ONLINE INTERFACE
COMM	COMMUNICATIONS	OPERI	OPERATOR INTERFACE
CNTRL	CONTROL, CONTROLLER	PROC	PROCESSING ROUTINES
DATA	DATA REDUCTION	SECUR	SECURITY
DBASE	DATA BASE	SEN	SENSOR SUPPORT ROUTINES
DIAG	DIAGNOSTICS	SIM	SIMULATION
DSPLY	DISPLAY	STASTO	START, STOP ROUTINES
ERR	ERROR CHECKS	STATUS	STATUS, HISTORY
EXEC	EXECUTIVE FUNCTIONS	TEST	TEST, TESTING
GUID	GUIDANCE, GUIDANCE AND CONTROL	TLM	TELEMETRY
INOUT	INPUT, OUTPUT	TIME	TIME/TIMING FUNCTIONS
MATH	MATHEMATICAL FUNCTIONS	TRK	TRACKING, POINTING, TARGETING
MNVR	MANEUVER, ATTITUDE/STATION,ETC.	UTIL	UTILITIES

Note: C = Category, E = Empirically derived.

SSM

Identification Inputs

1. **Organization** [S, No default, Required] - company, agency, or university for which the estimate is being made.

2. **Project name** [S, No default, Required] - name of the project for which the estimate is being made.

3. **File name** [S, No default, Required] - user assigned file name that contains the set of data used to derive a project estimate.

Modular (functional) components

1. **Module name** [S, No default, Required] - name associated with each defined software unit or function. Modules are defined at the discretion of the user. The key criteria of module definition is that a user must be able to discriminate the modules with respectto size. Number of modules > 3. The upper limit to the number of modules is constrained by available computer memory.

2. **Module description** [S, No default, Optional] - brief summary of the key operational/functional characteristics and requirements of the module.

3. **Reference module size** [E, No default, Required for at least two modules] - actual module size in the unit desired for SSM output. Size must be included for at least two of the defined modules. Reference modules do not necessarily have to be adopted into the new system being sized.

Note: Items 5 through 8 refer to relative ranking data provided by the user for all new modules.

5. **Pairwise data** [E, No default, Required] - from a unique pairing of all the modules in the project, judgements of which is the larger of the two for each pair.

6. **Ranking data** [E, No default, Required] - rank order of modules from largest to smallest.

7. **Sorting data** [E, No default, Required] - association of each module to a size interval. Size intervals are defined according to anticipated range of module sizes.

8. **Pert sizing data** [E, No default, Required] - for each module, in SLOC:

   a) lowest possible size
   b) most likely size
   c) highest possible size.

Note: S = Character string, E = Empirically derived.

APPENDIX B
HARDWARE CONFIGURATION

1. **ASSET-R**: ASSET-R runs on an IBM PC, XT, AT, or compatible with a minimum of 256K bytes of memory and a color or monochrome display. The system requires PC-DOS or MS-DOS, version 2.0 or higher. A minimum of one floppy disk drive is required. A hard disk drive and printer are optional.

2. **BYL**: BYL runs on an IBM PC, XT, AT, or compatible with a minimum of 512K bytes of memory and an 80-column, 24 or 25 line color or monochrome monitor. The system requires PC-DOS or MS-DOS, version 2.10 or higher. Two floppy disk drives, a floppy disk drive and hard disk drive, or a single IBM AT-type high capacity disk drive are required. A clock-calendar or multi-function board (such as, AST Six-Pack) and a printer are optional.

3. **CEIS**: CEIS runs on an IBM PC, XT, AT, compatible, or Zenith 100 with 256K bytes of memory and a color or monochrome display. The system requires PC-DOS or MS-DOS, version 2.0 or higher. A minimum of one floppy disk drive is required. CEIS can also be installed on a VAX 11/780 operating under VMS. In addition, CEIS can be accessed via a time-sharing system with an office terminal and standard modem.

4. **ESD**: ESD runs on a Zenith 100 with a color or monochrome display. A minimum of one floppy disk drive and a printer are required.

5. **PRICE SZ**: PRICE SZ runs on a PRIME minicomputer operating under PRIMOS. In addition, PRICE SZ can be accessed via a time-sharing system with an office terminal and standard modem.

6. **QSM SIZE PLANNER**: QSM SIZE PLANNER runs on an IBM PC, XT, AT, or compatible with 128K bytes of memory and a color graphics display. The operating system is DOS. Two floppy disk drives or a floppy disk drive and a hard disk drive are required.

7. **SIZER/FP**: SIZER/FP runs on an IBM PC, XT, AT, or compatible with 512K bytes of memory and a color or monochrome display. Two floppy disk drives or a floppy disk drive and a hard disk drive are required.

8. **SSA**: SSA runs on an IBM PC, XT, AT, or compatible with a color or monochrome display. The operating system is DOS. A minimum of one floppy disk drive is required.

9. **SSM**: SSM runs on an IBM PC, XT, AT, or compatible and a color or monochrome display. The operating system is DOS. A minimum of one floppy disk drive is required. A mainframe version of SSM also runs on a PRIME minicomputer operating under PRIMOS. Commercial users can access the mainframe version via a time-sharing system with an office terminal and standard modem.

APPENDIX C
CONTRACTUAL ARRANGEMENTS AND COSTS

1. **ASSET-R**: An ASSET-R annual licensing fee for one unit costs $8,000. This includes user support via telephone. An initial training session will be made available to users. However, at this time the cost for this training and whether the training will be mandatory have not been determined. Other licensing arrangements available are perpetual, corporate, and site. These agreements should be negotiated with Reifer Consultants since the term and number of units are variable. There is also a short term licensing arrangement available at $1,000 per month with a three month minimum and these fees are applicable towards a longer term licensing arrangement.

2. **BYL**: A BYL one time licensing fee for purchasing one unit costs $950. The cost for purchasing additional units decreases depending upon the number of units.

3. **CEIS**: CEIS is available to System-3 (software cost model) users at no additional cost. A licensing arrangement for non-System-3 users has not been established at this time. The System-3 (including CEIS) annual licensing fee for one unit is $9,400; additional units are $600 per year. This price includes a telephone help line and system upgrades at no additional charge. CEIS is available on a time sharing basis at $49.25 per hour of connect time from Computer Economics, Inc. (CEI). A dial-up to Wright Patterson AFB is also available for government users (DoD and NASA); contact Lt. Paul Marsey (513/244-6347) at Wright Patterson. System-3 has a three day training course available. This course is strongly recommended and costs $750 per person when given at the CEI facility. Government users may contract to have the course presented at their site to an unlimited number of students for a fee of $3,800 plus travel and living expenses for CEI personnel.

4. **ESD**: ESD will be available to anyone after September 1987 at no cost from ESD/ACCR at Hanscom AFB. A 5 1/4 " floppy disk should accompany the request.

5. **PRICE SZ**: PRICE SZ for software sizing is part of a package of PRICE family models which includes the PRICE S model for software costs and schedules and PRICE SL for software life cycle costs (maintenance, enhancement, and growth). Government users can use the PRICE S package on a time sharing basis at $75 per hour of connect time by contacting Lt. John Jones of the Aeronautical Systems Division at Wright Patterson AFB. Commercial users can use the PRICE S package on a time sharing basis at $13 per hour of connect time by contacting RCA PRICE Time-sharing Services at Moorestown, New Jersey, BUT they must also pay an access fee of $38,500 for one unit or $61,600 for multiple units. Commercial users can also lease the PRICE S package for installation on their own PRIME minicomputer at $58,500 per year for one unit or $81,600 per year for multiple units. A one week training course is mandatory and costs $1,125 for a government student or $1,500 for a commercial student. These costs include refresher training, manual updates, and technical assistance at no additional charge.

6. **QSM SIZE PLANNER**: A QSM SIZE PLANNER annual licensing fee for one unit costs $9,500; additional units are $500 each. This includes product upgrades, phone-in consultation, and newsletter support. A two day training course is mandatory for at least one user per site and costs $600 per student. Arrangements may be made to have the course presented at the user site. The cost for this is between $3,000 and $5,000 (plus travel and living expenses for the instructors) depending on number of students, hardware availability, etc.

7. **SIZER/FP**: A SIZER/FP one time licensing fee for purchasing one unit costs $500. The cost for purchasing additional units decreases depending upon the number of units.

8. **SSA**: SSA is available at no cost to qualified, government users from SD/ACCE at Los Angeles Air Force Station; contact Mr. Gerard Heydinger.

9. **SSM**: A SSM one time licensing fee for purchasing one unit costs $1849. The cost for purchasing additional units decreases depending upon the number of units. Commercial users can also access the mainframe version of SSM on a time sharing basis at $13 per hour of connect time by contacting RCA PRICE Time-Sharing Services at Moorestown, New Jersey.

APPENDIX D

DERIVATION OF FUNCTION POINT PARAMETERS FOR THE CATSS SENSITIVITY MODEL

Initial application of function point models to the CATSS Sensitivity Model yielded size estimates that were off from the actual system size by a large factor. Results motivated a review of function point input parameters to ascertain if definitions were applied incorrectly. With additional consultation, internal to IITRI, personnel were unable to definitively note inaccuracies in the initial application that would drastically reduce relative error. Additional sources of instruction were sought which resulted in the enrollment of IITRI personnel in the Function Point Analysis Workshop conducted by IBM Information System Services. The workshop, though geared toward data processing systems, was extremely useful in understanding how to apply the parameter definitions.

After the workshop, IITRI personnel discussed the CATSS Sensitivety Model with the workshop instructor who was shown sample inputs, outputs, selection menus, and data files of the CATSS system. The ensuing discussion was enlightening and revealed where IITRI had incorrectly identified FP parameters by pointing out guidelines that were overlooked.

The key to counting function point parameters is consistency. A set of examples representative of common situations together with the basic definitions set a pattern of how function points are counted. If an analyst comes across a situation in which ambiguity exists relative to counting, current guidelines and definitions should be used to decide. How the situtation was handled should be made known on a broad basis and, thereafter, followed when the same situation occurs.

This appendix contains a description of how function point parameters for the CATSS Sensitivity Model were identified. Counting conventions are presented by parameter type:

- External Inputs
- External Outputs
- Logical Internal Files
- External Inquiries
- External Interfaces.

Illustrations will be included in the descriptions for clarity. Also, items that were overlooked in the initial application will be noted.

It is recommended that the reader review Section 1.2 which is an overview

of the CATSS Sensitivity Model. Also, a basic knowledge of function points is assumed.

This appendix does not focus on the complexities of the elements counted (discriminants required for the BYL and QSM function point implementations).

For each parameter type the description of how counts were derived contains the following information:

- **Key points** - a summary of the basic parameter definition with emphasis on certain key factors.
- **Potential types within the CATSS Sensitivity Model** - situations in which elements were counted as this parameter type.
- **Description** - an explaination of how the parameter count was obtained as illustrated by a few representative examples. Also noted are some of the early mistakes that were made relative to identifying parameters.
- **Total number of element types** - total count of elements for the specified parameter.

EXTERNAL INPUTS

Key Points

- An input is user data or user control information that enters the external boundary of the applicaton.
- It must change something inside the system.
- It is unique if it has a different format or requires different processing logic.

Potential types within the CATSS Sensitivity Model

- Input screens for data entry
- PF1 key
- Threat or target location designation
- Flight path location designation

Description - When the select threat location option is designated for the TA function, a terrain contour map is displayed with several high points designated for assistance in point selection. The threat location is selected by moving the cursor to any desired point in the area and pressing the carriage return key. A 5 x 5 grid point area centered at the point indicated by the cursor is searched for the highest terrain point, and this is used as the candidate threat location. An identical process is followed for target selection in the R/S function. Threat or target location designation, flight path location designation (described in Section 4.2.2), and the PF1 key which controls main and subsystem selection menu displays are counted as one input each.

Screens for data entry are the final type of input. Two screens that are of the same format and provide data or control information for the same processing task are counted as one input. The following example illustrates how input screens are counted. The Threat Avoidance function consists of the four input screens shown in Figure D-1. The DATABASE MENU provides user data for elevation degradation. The AIRCRAFT PARAMETERS MENU provides input used to calculate the aircraft's profile along the designated path. The DISPLAY PARAMETERS MENU and the THREAT PARAMETERS MENU solicit information used for visibility computation. Hence, the number of input screens for the TA function are three (3). The same logic applies for counting input screens or other subsystem functions.

DISPLAY PARAMETERS

___ Range Step (m) (100.00)
___ Angle Step (degrees) (10.00)
___ Aircraft Altitude (m AGL) (60.00)

PF1 key to Return to Threat Avoidance Menu

AIRCRAFT PARAMETERS MENU

___ Aircraft Ground Speed (m/s) (250.00)
___ Terrain Following Height (m AGL) (60.00)
___ Maximum Vertical Acceleration (g's) (1.50)
___ Maximum Vertical Deceleration (g's) (-.75)
___ Aircraft Turn Limits (g's) (2.00)

PF1 key to Return to Threat Avoidance Menu

DATA BASE MENU

___ Correlation Coefficient (0.92)
___ Mean of Deviations (m) (bias) (0.00)
___ Standard Deviation (m) (7.50)

PF1 key to Return to Function Menu

THREAT PARAMETERS MENU

___ Altitude (m AGL) (10.00)
___ Range (km) (50.00)
___ Select Threat Location

PF1 key to Return to Threat Avoidance Menu

Figure D-1. Input Screens for the CATSS Threat Avoidance Function.

Total Number of CATSS External Inputs

PF1 key:	1
Threat or target selection:	1
Flight path selection:	1
Input screens:	11
TOTAL:	14

EXTERNAL OUTPUTS

Key Points

- An output is user data or user control information that leaves the external boundary of the application measured.
- It is unique if it has a different format or requires different processing logic.
- It does not include output response of an external inquiry.
- It does not include output files of records.
- It should not be assumed that a gerneated report or graphic display is always one output.

Potential types within the CATSS Sensitivity Model

- Graphic displays
- Generated reports written to a file on disk
- Smoothed flight path
- Radial lines emanating from an antenna site showing coverage area
- Statistical summaries

Description - A key consideration to counting external outputs is not assuming that a summary report or graphic display is always counted as a single outout. The following representative sample of CATSS output demonstrates this point.

Threat Avoidance Function Output

The next three figures are output generated by the Threat Avoidance function. Figure D-2 shows a contour map of an area with four threat locations and a designated flight path. The number of external outputs requiring different processing logic presented in Figure D-2 is two. One output is the flight path which is smoothed at the corners according to selected lateral acceleration and aircraft speed. The second output is the coverage region of each threat as determined by line-of-sight calculation. The terrain contours are not designated as a third external output because the contours were in the proper format for display and did not require additional processing. Also, the contours are generated as an output response to an external inquiry type and are counted as such.

In Figure D-3, the status of the Threat Avoidance function along the path is displayed on the screen. In the upper margin of the display are shown the system parameters selected for the run. The central portion shows the position of the aircraft along the path. The visibility from each threat to

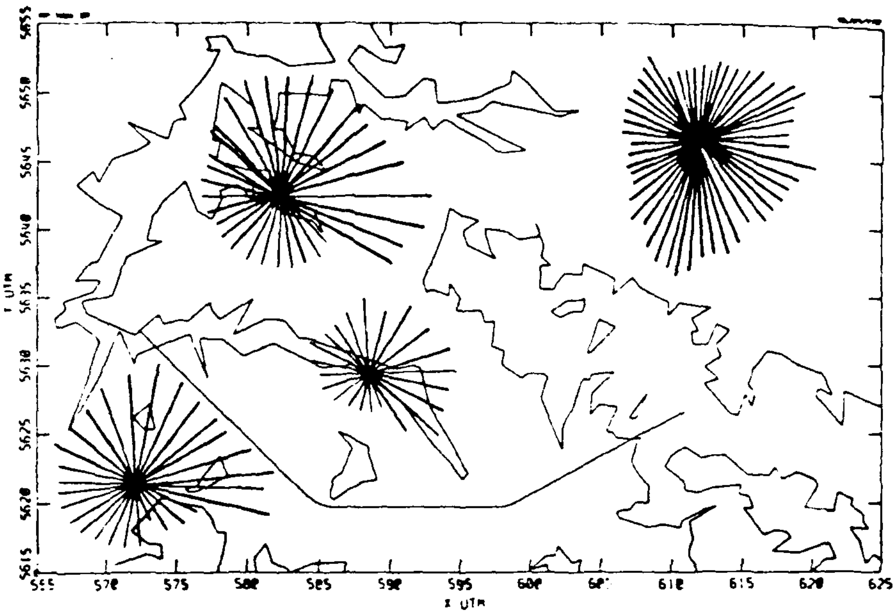


Figure D-2. Threat Avoidance Path with Four Threats. The Number of External Outputs Exhibited is Two.

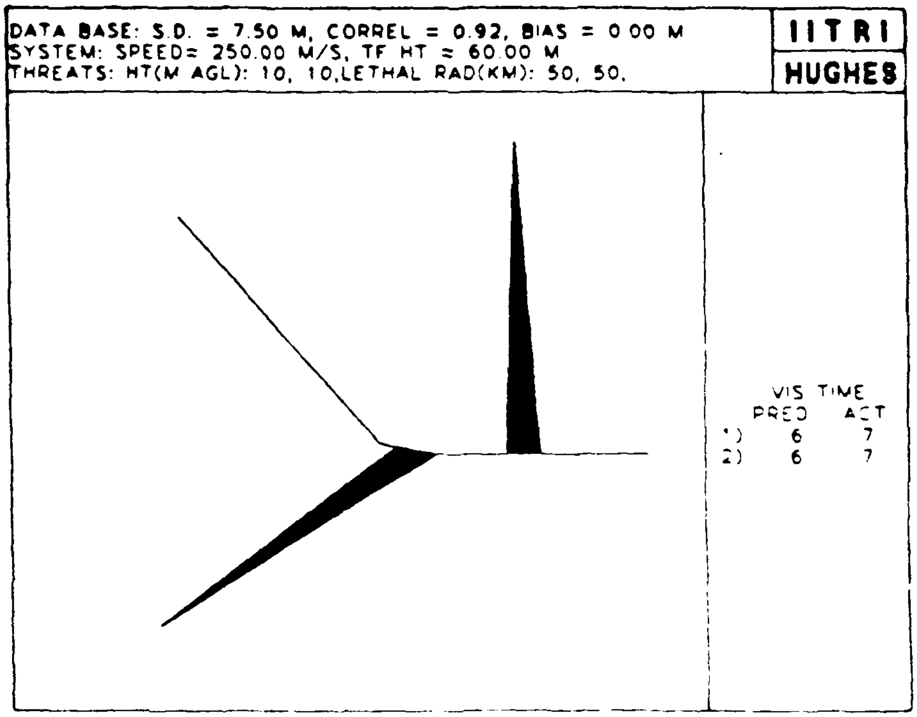


Figure D-3. Threat Avoidance Function Display Showing Visibility from Two Threats to the Aircraft Along the Path. The Number of External Outputs Exhibited is Two.

the aircraft is calculated at each point along the path. If visibility exists, a line is drawn from the threat to the aircraft. Shown in the right margin are the accumulated visibility values. Figure D-3 is the result of two unique processing tasks. One generated the central portion of the graphic. The other accumulated visibility values shown in the right margin. The system parameters shown in the upper margin are not calculated values. They are essentially a reformatted display of user input and were already counted as such.

When the run terminates, the threat visibility summary in Figure D-4 is generated. It shows statistics of the predicted and actual visibility of the aircraft from each threat. It is counted as one external output.

Cross-Country Movement Function Output

To understand how CCM external outputs are counted, it is necessary to review some underlying concepts. The basis for CCM analysis is the calculation of vehicle speed within an area of known natural features. CCM analysis examines the influence of slope, vegetation, soil, and surface roughness features on the maximum speed of a selected vehicle across a designated area. Each feature is used to calculate a value between 0 and 1, and the product of the four feature values and the vehicle's maximum speed is used to compute the vehicle speed within an area. A speed grid is built using the computed speeds. In addition the speed grid is converted into a grid with integer values indicating speed categories: go, slow-go, and no-go. The integer speed category grid is used to create the actual cross-country movement maps - one from true data and one from degraded data - providing a visual tool in determining the effects of degraded data on CCM analysis.

Figure D-5 was viewed as exhibiting one external output. The categorization of data into three speed categories was part of one processing task. A separate software module was used to generate the legend in the right margin. However, the information in the legend is not the result of change in internal data stores or logical internal files.

Two CCM maps are output from CCM analysis: a true and degraded map indicating the data set used in the analysis. Both maps are counted as one external output since the processing to generate each map is the same.

Threat # 1 (611.8000, 5646.100)

		ACTUAL VISIBLE	ACTUAL MASKED	
PREDICTED	VISIBLE	0	1	1
	MASKED	7	424	431
		7	425	432

Threat # 2 (582.0000, 5646.600)

		ACTUAL VISIBLE	ACTUAL MASKED	
PREDICTED	VISIBLE	102	8	110
	MASKED	1	321	322
		103	329	432

Threat # 3 (588.4000, 5629.300)

		ACTUAL VISIBLE	ACTUAL MASKED	
PREDICTED	VISIBLE	0	0	0
	MASKED	0	432	432
		0	432	432

Threat # 4 (572.0000, 5621.400)

		ACTUAL VISIBLE	ACTUAL MASKED	
PREDICTED	VISIBLE	118	0	118
	MASKED	2	312	314
		120	312	432

Figure D-4. The Threat Visibility Summary is Counted as One External Output.

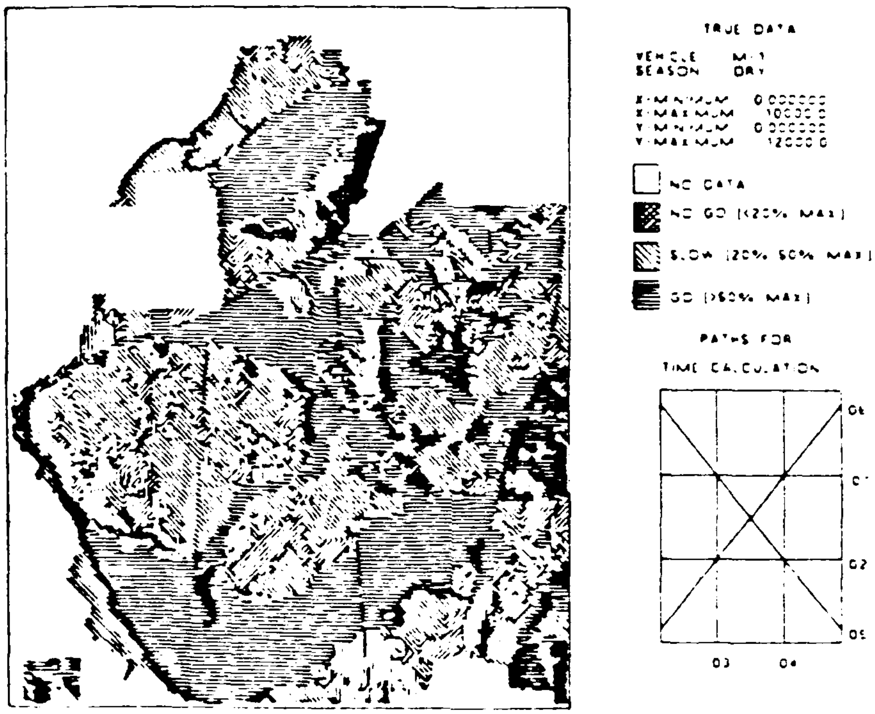


Figure D-5. The Cross-Country Movement Map Counts as One External Output.

Two unique processing tasks are required to generate the CCM report file in Figure D-6. A set of paths across the map area are examined and traversal times are calculated in one task. Different processing logic is required to generate the set of statistics at the end of the CCM report (standard deviation, average difference, kappa, etc.) which are computed by performing cell-by-cell comparisons between degraded and nondegraded speed grids. The top portion of Figure D-6 summarizes the input parameters used to generate the speed grids and traversal times. Hence, the CCM report file counts as two external outputs.

Several distribution plots are created during CCM runs using VAX DECgraph. DECgraph is a software tool for generating graphs from data. DECgraph generated plots are not counted as external output. The CCM function also provides a simple file manipulation tool that creates a new file from existing files by deleting or merging records. New files created in this type of operation are not counted as external output.

CROSS-COUNTRY MOVEMENT

COORDINATES OF AREA MAPPED:

X MINIMUM VALUE:	1000.00	X MAXIMUM VALUE:	3012.50
Y MINIMUM VALUE:	0.00	Y MAXIMUM VALUE:	2000.00

INPUT PARAMETER	VALUE
VEHICLE TYPE	FOOT
SEASON	SUMMER DRY
MAXIMUM ON ROAD GRADABILITY (%)	100.00
MAXIMUM OFF ROAD GRADABILITY (%)	100.00
MAXIMUM SPEED (KMH) OF VEHICLE	6.40
WIDTH (METERS) OF VEHICLE	0.45
TURNING RADIUS (METERS) OF VEHICLE	0.61
MAXIMUM SHEARABLE STEM DIAMETER (METERS)	0.01
VEHICLE CONE INDEX FOR 1 PASS	5.00
VEHICLE CONE INDEX FOR 50 PASSES	60.00

DATA SET DEGRADATION PARAMETERS

DEGRADE FACTOR	DEGRADE METHOD	DEGRADE PARAMETER	RECATEGORIZING PARAMETER
STEM SPACING	DOMINANT	10	NO RECATEGORIZE
STEM DIAMETER	DOMINANT	10	NO RECATEGORIZE
VEGETATION ROUGHNESS	DOMINANT	10	NO RECATEGORIZE
SURFACE ROUGHNESS	DOMINANT	10	NO RECATEGORIZE
SOIL RATED CONE INDEX	DOMINANT	10	NO RECATEGORIZE
SLOPE	DOMINANT	10	NO RECATEGORIZE

PATH	TRAVERSAL TIME (HOURS)	RATIO WITH TRUE
1	0.437	1.000
2	0.437	1.000
3	0.451	1.030
4	0.428	0.991
5	0.605	1.000
6	0.614	1.000

STANDARD DEVIATION (KM/H)	=	0.4
AVERAGE DIFFERENCE (KM/H)	=	0.0
MAXIMUM CHANGE (KM/H)	=	4.5
KAPPA	=	0.797
FRACTION OF SAME COLOR CELLS	=	0.988

Figure D-6. The Cross-Country Movement Report File Exhibits Two External Outputs.

Total Number of CATSS External Outputs - A total of 14 external outputs are generated by the CATSS Sensitivity Model. Generally, each report or graphic display was counted as a single external output. Instances in which a generated report or display exhibited more than one external output parameter type are discussed in the preceeding description.

LOGICAL INTERNAL FILES

Key Points

- A logical internal file is each logical group of data that is generated, used, and maintained by the application.
- Logical internal files are accessible to the user through external input, output or inquiry types.
- Databases are logical internal file types.
- Each hierarchical path through a database, derived from user requirements counts as a single logical internal file type.
- The user must be aware that the file exists.

Potential types within the CATSS Sensitivity Model

- Terrain elevation data
- Coordinates of elevation contour lines
- Road data
- Areal feature data
- Linear feature data
- Vehicle parameters internal array
- Default parameters file

Description - Five databases are used in the CATSS Sensitivity Model: one contains terrain elevation data, one contains coordinates of elevation contour lines, one contains road data, one contains areal feature data, and one contains linear feature data. Figure D-7 illustrates the relationship of these databases to the subsystem

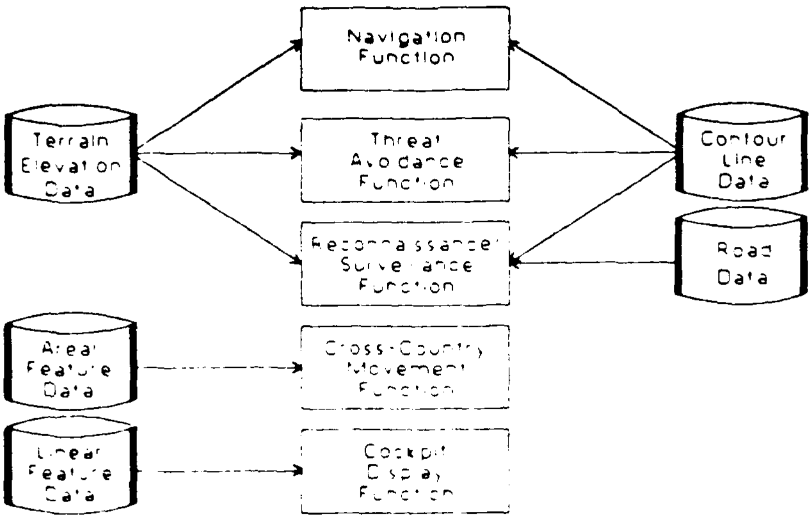


Figure D-7. Databases Used by the CATSS Sensitivity Model.

functions. Note from the figure that they are read-only files and are not generated or maintained by the system. Each is counted as one logical internal file.

Two other entities were counted as logical internal file types. One is a list of vehicle parameters required for CCM analysis shown in Table D-1. The CCM program contains an internal array of eight (8) vehicle types with pre-defined parameters. The user may select one of these, or specify a new vehicle type with other parameters. There is no capability to add a new vehicle type to the internal array.

TABLE D-1

VEHICLE PARAMETERS REQUIRED FOR CCM ANALYSIS

MAXIMUM ON-ROAD GRADABILITY (%)
MAXIMUM OFF-ROAD GRADABILITY (%)
MAXIMUM SPEED (KM/H)
WIDTH (METERS)
TURNING RADIUS (METERS)
MAXIMUM SHEARABLE STEM DIAMETER (METERS)
VEHICLE CONE INDEX -- ONE PASS
VEHICLE CONE INDEX -- 50 PASSES

Another logical internal file type is a default parameters file which initializes many of the system variables in common. CATSS input screens are displayed showing each system parameter along with its default value. The user may change the value through key-in, however, this will not affect the default parameters file.

Total Number of CATSS Logical Internal Files - A total of 7 logical internal files were counted for the CATSS Sensitivity Model.

EXTERNAL INQUIRIES

Key Points

- An inquiry is each unique input/output combination.
- It causes no change to internal data.
- It causes and generates an immediate output.

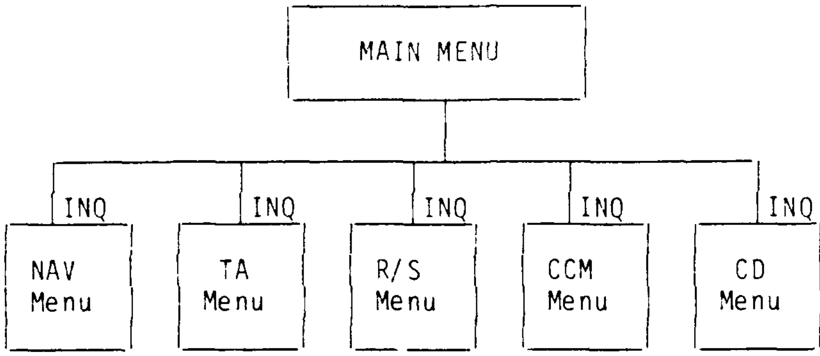
Potential types within the CATSS Sensitivity Model

- Selection menu screens

Description - Initially, IITRI personnel counted each selection menu screen that received input and generated output as one inquiry; even if, depending upon the selected option, the generated output was of a different format and type. The IBM workshop instructor [IBM87] eluded to guidelines and suggested that each option of the selection menu screen that generates a unique output should be counted as a separate inquiry. The correct approach is demonstrated in Figure D-8, diagram (a), for the top-level main selection menu and generated subsystem level menus. Part (b) of Figure D-8 shows how inquiries were initially counted.

Figure D-9 shows how inquiries are counted for the Navigation Function. Each input/output combination for the Navigation subsystem function are illustrated in diagram (a) of Figure D-9. The Degrade Data Base Option selection of the NAVIGATION MENU generates the DATA BASE MENU. The Specify System parameters option of the NAVIGATION MENU generates the SYSTEM PARAMETERS MENU. The SYSTEM PARAMETERS MENU will in turn generate the INS ERROR MENU. A contour map is automatically displayed when the analyst selects the Choose Path option from the NAVIGATION FUNCTION MENU.

(a)



(b)

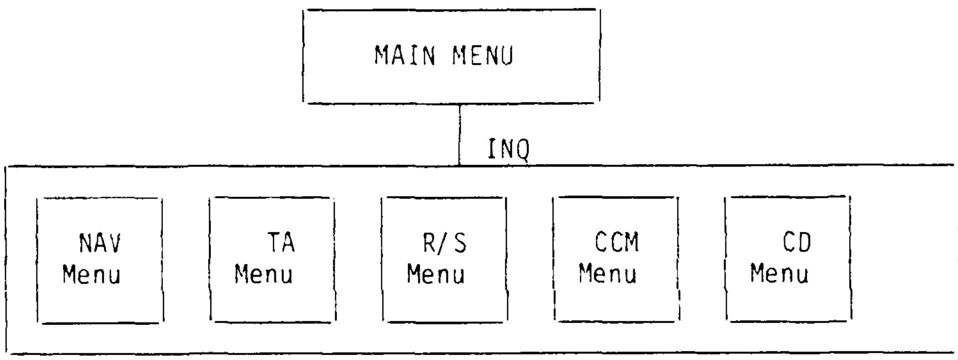
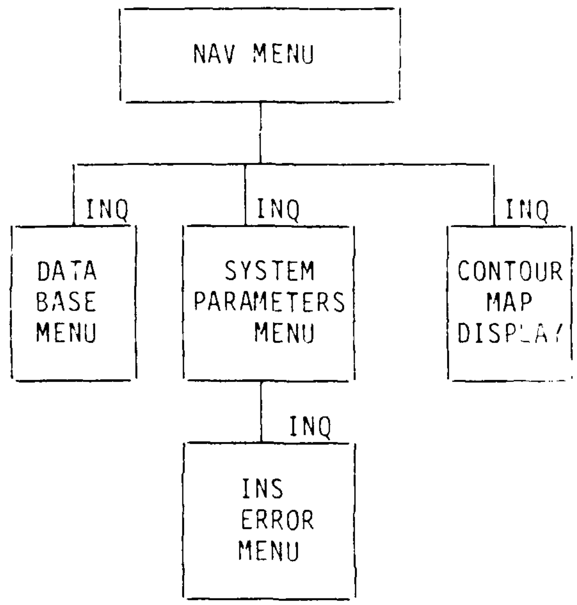


Figure D-8. Depending Upon the Option Selected from the Main Menu, a New and Different Selection Menu is Generated at the Subsystem Level. Five Inquiries are Counted for Each Unique Input/Output Pair in Diagram (a). Originally, this was Incorrectly Viewed as One Inquiry as Shown in (b).

The CCM and CD subsystem functions generate, merge, display and print reports and plots at the users option whereas the NAV, TA, and R/S functions do not. Figure D-10 illustrates CCM options.

(a)



(b)

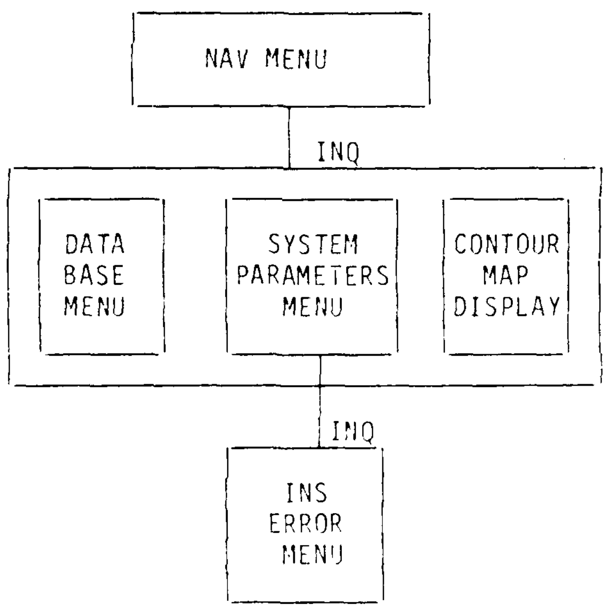


Figure D-9. There are Four Unique Input/Output Pairs in the Navigation Function Shown in Diagram (a). Originally, this was Incorrectly Viewed as Two Inquiries as Shown in Diagram (b).

CROSS-COUNTRY MOVEMENT MENU
____ CCM Runs
____ Generate Summary Report
____ Generate Summary Plot
____ Display Existing Reports
____ Print Existing Reports
____ Delete/Merge Results
____ Exit

Figure D-10. The Main Menu for the Cross-Country Movement Function.

The first option, CCM Runs, generates a series of input/selection screens much like the NAV and TA functions described in Section 4.2.2.

The remaining options, generally prompt the user to enter the name of the file to process and/or the name of the new file to generate. These options are not counted as inquiries since generated reports are counted as output. Also, the output or display of existing reports is not counted.

Total Number of Inquiries

Both function point counts; the corrected values (a) contrasted with those initially obtained (b) are provided.

	(a)	(b)
Main Menu to Subsystem Menu:	5	1
NAV:	4	2
TA:	5	2
R/S:	7	2
CCM:	4	2
CD:	2	2
TOTAL:	27	11

EXTERNAL INTERFACES

Key Points

- Files passed or shared between applications should be counted as external interface types.

Potential types within the CATSS Sensitivity Model

- Results file used to generate DECgraph plots

Description - An output of CCM and CD analysis is a results file that can be used to generate summary plots through a DECgraph interface. The results file is counted as one external interface.

Total number of External Interfaces - A total of 1 external interface was counted for the CATSS Sensitivity Model.

APPENDIX E

ADDITIONAL SOURCES FOR FUNCTION POINT TRAINING

One of the disadvantages of function points has been substantial ambiguity in exactly how to define and count the function point parameters. Permanent study groups within both IBM and GUIDE have put forth an intensive effort to refine existing definitions of function points so that independent companies and individuals would get the same results when analyzing the same project [ZWAN84]. (The GUIDE Project was formed in 1982. It is part of the Data and Productivity Management Division of IBM.) There are at least two comprehensive documents whose purpose is to explain how to perform the methodology and how to use the estimates:

1. Ken Zwanzig, ed., Handbook for Estimating Using Function Points, GUIDE International, November, 1984.
2. A.J. Albrecht, ADM Productivity Measurement and Estimate Validation, CIS & A Guidelines 313, November, 1984.

IBM Information Services offers a one-day Function Point Analysis Workshop. The workshop size is limited to 20 attendees and is conducted on IBM or customer premises. To enroll, either call an IBM Information Services representative, or call the IBM education number (1-800-IBM-2468) to find out various locations where workshops are held. The course number is WS170 and cost is $350 per attendee. IITRI personnel who have attended the workshop found it very beneficial though geared toward data processing systems. The workshop provides a case study in counting function points and in addition, demonstrates their use at IBM to evaluate productivity and quality for development efforts. The workshop does not address the use of function points to derive a SLOC estimate which is a primary focus in this report.

REFERENCES

[AERO85a] The Aerospace Corporation, Software Sizing Research Effort, September 1985.

[AERO85b] The Aerospace Corporation, Software Sizing Capability, May 1985.

[AFSC87] Air Force Systems Command, Software Risk Management, AFSC Pamphlet 800-XX, June 1987.

[ALBR83] Albrecht, Allan J. and Gaffney Jr., John E., "Software Function, Source Lines of Code, and Development Effort Prediction: A Software Science Validation", IEEE Transactions on Software Engineering, November 1983, pp. 639-648.

[ALBR84] Albrecht, A.J., AD/M Productivity Measurement and Estimate Validation, CIS & A Guideline 313, November 1984.

[ARINC85] ARINC Research Corporation, Software Sizing and Cost Estimation, ARINC Research Publication 3243-01-TR-3675, July 1985.

[BOEH81] Boehm, Barry W., Software Engineering Economics, Prentice-Hall Englewood Cliffs, NJ, 1981.

[BOZO84] Bozoki, George J., SSM User's Guide, GJB Associates, 1984.

[BOZO87] Bozoki, George J., Target Software, May/June, 1987.

[BRIT85] Britcher, R.N. and Gaffney Jr., J.E., "Reliable Size Estimates for Software Systems Decomposed as State Machines", Proceedings of IEEE COMSAC 85, 9-11 October 1985.

[CATS86a] CATSS Sensitivity Model, Functional Description, IIT Research Institute (IITRI) and Hughes Aircraft Company, 28 March 1986.

[CATS86b] CATSS Sensitivity Model, Sensitivity Analysis Modules User Manual, IITRI and Hughes, 31 October 1986.

[CATS86c] CATSS Sensitivity Model, Sensitivity Analysis Modules Program Mainentance Manual, IITRI and Hughes, 31 October 1986.

[CATS86d] CATSS Sensitivity Model, Technical Report, IITRI and Hughes, 31 October 1986.

[DEAN83] Dean, Joseph P., "Estimating Lines of Code at the Air Force Communication Computer Programming Center", Fourteenth Annual USAF Academy Computer - Related Information Systems Symposium (CRISYS-83), AFCCPC/SKXY-A, 1983.

[DEAN87] Dean, Joseph P., ESD/ACCR, Hanscom AFB, March 1987.

[ESD86] The ESD Software Sizing Data Base, ESD/ACCR, Hanscom AFB, March 1986.

[GAFF84] Gaffney Jr., John E., "Estimation of Software Code Size Based on Quantitative Aspects of Function (With Application of Expert Systems Technology)", Journal of Parametrics, Vol. 4, No. 3, September 1984.

[GORD87] Gordon Group, Before You Leap User's Guide, 1987.

[LAMB86] Lambert, Joseph M., "A Software Sizing Model", Journal of Parametrics, Vol. VI, No. 4, December 1986.

[MANC87] Mancini, Kenneth M., Manager, Eastern Regional Sales, RCA PRICE Systems, March 1987.

[MENT87] Mentzel, Barbara, ESD/ACCR, Hanscom AFB, July 1987.

[MIZE87] Mizer, David, Resource Cost Analysis Office, Architecture Planning and Technology Division, The Aerospace Corporation, May 1987.

[PUTN78] Putnam, Lawrence H., "Example of an Early Sizing, Cost and Schedule Estimate for an Application Software System", Proceedings COMPSAC78, 13-16 November 1978.

[PUTN87] Putnam, Douglas T., Director of Marketing, Quantitative Software Management, Inc., May 1987.

[QSM87] Quantitative Software Management, Inc., Size Planner Manual, 1987.

[RAPP85] Rapp, William, "RCA PRICE Parametric Software Sizing - A State-of-the-Art Approach", Proceedings of the International Society of Parametric Analysts Seventh Annual Conference, Vol. 4, No. 1, 7-9 May 1985.

[RCA85] RCA Corporation, PRICE SZ Reference Manual, 1985.

[RCI87] Reifer Consultants, Inc., ASSET-R User's Manual, RCI-TN-270-UM (V1.0), June 1987.

[REIF86] Reifer, Donald J. "Predicting the Size of Real-Time Systems", Proceedings of the NIS Software Cost and Quality Management Conference, 29-30 October 1986.

[REIF87] Reifer, Donald J. Analytical Software Size Estimation Tool - Real-Time (ASSET-R): An Overview, RCI-TN-113, May 1987.

[SAAT80] Saaty, Thomas L., The Analytical Hierarchy Process, McGraw Hill, NY, 1980.

[SAP] SAP User's Manual

[SPR86a] Software Productivity Research, Inc. (SPR), SPQR/20 User's Guide, 1986.

[SPR86b] SPR, Software Productivity, Vol. I, No. 1, March/April 1986.

[SPR87] SPR, The SPR Feature Point Method (Briefing materials), 1987.

[ZWAN84] Zwanzig, Ken, eg., Handbook for Estimating Using Function Points, GUIDE International, November 1984.